P515416.PDF [Page: 3 of 97]

Copy No. 1

DEFENCE R&D DÉFENSE

Incorporation of UK NS94 Ultimate Strength Analysis in MGDSA

Part II: Software Considerations

M. Lichodzijewski

L. Jiang

MARTEC Ltd.,
Suite 400, 1888 Brunswick Street,
Halifax, NS B3J 3J8

DEFENCE RESEARCH ESTABLISHMENT ATLANTIC

Contractor Report

DREA CR 2000-113

March 2000

National Defence Défense nationale

Canada

THIS IS AN UNEDITED REPORT ON SCIENTIFIC OR TECHNICAL WORK CONTRACTED BY THE DEFENCE RESEARCH ESTABLISHMENT ATLANTIC OF DEFENCE R&D CANADA.

THE CONTENTS OF THE REPORT ARE THE RESPONSIBILITY OF THE CONTRACTOR, AND DO NOT NECESSARILY REFLECT THE OFFICIAL POLICIES OF THE DEPARTMENT OF NATIONAL DEFENCE.

PLEASE DIRECT ENQUIRIES TO:

THE DIRECTOR GENERAL
DEFENCE RESEARCH ESTABLISHMENT ATLANTIC
P.O. BOX 1012
DARTMOUTH, NOVA SCOTIA, CANADA
B2Y 3Z7

Incorporation of UK NS94 Ultimate Strength Analysis in MGDSA

Part II: Software Considerations

M. Lichodzijewski

L. Jiang

MARTEC Ltd.,
Suite 400, 1888 Brunswick Street,
Halifax, NS B3J 3J8

Defence Research Establishment Atlantic

Contractor Report

DREA CR 2000-113

March 2000

P515416.PDF [Page: 6 of 97]

102-02177/

Authors

M. Lichodziiewski and L. Jiang

Approved by

Malcolm Smith

Contract Scientific Authority

Approved for release by

Walter E. Ellis

Chair, DREA Document Review Panel

© Her Majesty the Queen as represented by the Minister of National Defence, 2000

INCORPORATION OF UK NS94 ULTIMATE STRENGTH ANALYSIS IN MGDSA

PART II: SOFTWARE CONSIDERATIONS

Martec Report No.: TR-00-13

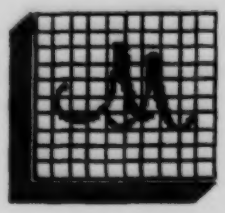
By:

M. Lichodzijewski
L. Jiang

Prepared for:

Defence Research Establishment Atlantic

Contract No.: W8477-8-0004/001/SV
PWGSC File No.: 005sv.W8477-8-0004



Martec Limited
Advanced Engineering and Research Consultants

400-1888 Brunswick Street
Halifax, NS B3J 3J8
Tel: (902) 425-5101
Fax: (902) 421-1923

101-717 boul St. Joseph
Hull, PQ J8Y 4B6
Tel: (819) 595-3526
Fax: (819) 595-1739

March 2000

ABSTRACT

The Ultimate Strength application is introduced. Its system architecture is described. A general overview of the representation of a cross section within the application is provided. Important aspects of the application are discussed. A bibliography and appendices are provided.

RÉSUMÉ

L'application d'analyse de la contrainte de rupture est présentée. L'architecture de son système est décrite. Une vue d'ensemble de la représentation d'une section transversale de l'application est fournie. Les aspects importants de l'application sont décrits. Une bibliographie et des annexes sont incluses.

INCORPORATION OF UK NS94 ULTIMATE STRENGTH ANALYSIS IN MGDSA PART II: SOFTWARE CONSIDERATIONS

DREA CR 2000-113

M. Lichodzijewski and L. Jiang
Martec Limited

EXECUTIVE SUMMARY

Introduction

Ultimate strength analysis is undertaken to find the bending load that will cause the collapse of a ship hull. Accurate knowledge of this is essential to determine a ship's weight limit and allowable operating conditions. Over the years, specialised analysis codes for evaluating ultimate strength have been developed by defence agencies in several countries. In Canada, the computer code ULTMAT was developed at DREA; in the UK, the NS94 suite of codes was developed at DERA Rosyth. Martec Limited of Halifax Nova Scotia, under contract to DERA Rosyth and DREA, was tasked with bringing together the UK and Canadian ultimate strength analysis codes in a single software application with a common user interface and a common database. A further goal was to provide a link between ultimate strength analysis and the finite-element based ship structural analysis program MGDSA (MAESTRO Graphics – Detail Stress Analysis). The overall objectives were to provide easier access to the analysis tools and to automate the ultimate strength modelling and evaluation process.

Principal Results

This report describes the aspects involved in the software development of the unified ultimate strength application. As a result of the developments undertaken in this work, cross-sectional models can be generated automatically from existing finite element ship models. Once this has occurred, an ultimate strength analysis can be carried out immediately through the application interface. Customisation of the model, data and analysis options is also permitted. Graphical tools are provided for reviewing of results. The net result is a very significant reduction in the time required to carry out ultimate strength modelling and analysis. With the unified application, an ultimate strength assessment can now be carried out in a matter of minutes if a finite element model of the ship already exists. This is a considerable increase in efficiency over previous efforts requiring several hours.

Significance of Results

Automated evaluation of ultimate strength is an essential component of the ongoing Improved Ship Structural Maintenance Management (ISSMM) project. The results of this study directly pertain to how ultimate strength analysis is to be implemented in the ISSMM Software Tool (IST). The automation and improved efficiency that has resulted from this work will mean that ultimate strength analysis will now be accessible to a wider group of users in the CF community. Furthermore, the Navy will be provided with answers to its ultimate strength queries more quickly, and hence a reduced cost.

INCORPORATION DANS MGDSA DE L'ANALYSE DE LA CONTRAINTE DE RUPTURE NS94 DU ROYAUME-UNI
PARTIE I : CONSIDÉRATIONS RELATIVES AU LOGICIEL

DREA-CR-2000-113

M. Lichodzijewski et L. Jiang
Martec Limited

SOMMAIRE

Introduction
Une analyse de la contrainte de rupture est entreprise dans le but de déterminer la charge de flexion qui provoquera la rupture d'une coque de navire. La connaissance précise de cette charge est essentielle pour établir le poids limite et les conditions d'exploitation admissibles d'un navire. Avec les années, des organisations de défense de plusieurs pays ont constitué des codes d'analyse spécialisés permettant d'évaluer la contrainte de rupture. Au Canada, le code informatique ULTMAT a été développé au CRDA; au Royaume-Uni, l'ensemble de codes NS94 a été établi au DERA Rosyth. Dans le cadre d'un contrat du DERA Rosyth et du CRDA, Martec Limited d'Halifax (Nouvelle-Écosse) a été chargée de regrouper les codes d'analyse de la contrainte de rupture du Royaume-Uni et du Canada afin de constituer une seule application logicielle à interface utilisateur commune et à base de données commune. Les travaux visaient également à constituer un lien entre l'analyse de la contrainte de rupture et le programme MGDSA d'analyse structurelle des navires à partir d'éléments finis (MAESTRO Graphics - Analyse détaillée de contrainte). Les objectifs globaux consistaient à faciliter l'accès aux outils d'analyse et à automatiser le processus de modélisation et d'évaluation de la contrainte de rupture.

Principaux résultats
Ce rapport décrit les aspects relatifs au logiciel que comporte le développement de l'application unifiée d'analyse de la contrainte de rupture. Les résultats des travaux effectués permettent de générer automatiquement des modèles de sections transversales à partir des modèles de navires existants à éléments finis. Cela fait, l'interface de l'application peut servir à exécuter immédiatement une analyse de la contrainte de rupture. Il est également possible de personnaliser le modèle, les données et les options d'analyse. Des outils graphiques sont prévus pour l'examen des résultats. Il est ainsi possible de réduire considérablement le temps nécessaire pour effectuer la modélisation et l'analyse de la contrainte de rupture. L'application unifiée permet maintenant d'effectuer une évaluation de la contrainte de rupture en quelques minutes seulement s'il existe déjà un modèle à éléments finis du navire. Cela représente une nette augmentation d'efficacité par rapport aux travaux antérieurs, qui prenaient plusieurs heures.

Importance des résultats
L'évaluation automatisée de la contrainte de rupture constitue un élément essentiel de l'actuel projet de gestion améliorée de la maintenance de la structure des navires (GAMSN). Les résultats de cette étude portent directement sur la façon dont l'analyse de la contrainte de rupture doit être intégrée à l'outil logiciel GAMSN (IST). L'automatisation et l'amélioration d'efficacité qui ont résulté de ces travaux signifient que l'analyse de la contrainte de rupture sera dorénavant accessible à un plus large groupe d'utilisateurs au sein des FC. La Marine recevra donc plus rapidement des réponses à ses questions de contrainte de rupture, ce qui entraînera une réduction des coûts.

TABLE OF CONTENTS

ABSTRACT I

TABLE OF CONTENTS II

LIST OF FIGURES V

LIST OF TABLES VI

1 BACKGROUND 1.1

1.1 INTRODUCTION 1.1
1.2 SYSTEM ARCHITECTURE 1.2
1.2.1 The Input Sub-system *1.3*
1.2.2 The Material Database *1.3*
1.2.3 The Damage Database *1.4*
1.2.4 The Beam Database *1.4*
1.2.5 The Load-Shortening Curve Database *1.4*
1.2.6 The Ultimate Strength Analysis Sub-system *1.6*
1.2.7 The Finite Element Analysis Sub-system *1.6*
1.2.8 The HOOD Core Sub-system *1.7*
1.2.9 The Main Shell Sub-system *1.7*

2 IMPLEMENTATION 2.1

2.1 THE REPRESENTATION OF A CROSS-SECTION 2.1
2.1.1 Geometric Elements *2.2*
2.1.1.1 Key-Points 2.2
2.1.1.2 Basic Structural Elements 2.3
2.1.1.2.1 Plates 2.3
2.1.1.2.2 Stiffeners 2.3
2.1.1.3 Structural Units 2.4
2.1.1.3.1 Transversely Stiffened Panels 2.5
2.1.1.3.2 Longitudinally Stiffened Panels 2.6
2.1.1.3.3 Hard Corners 2.8
2.1.1.3.4 User-Defined Structures 2.9
2.2 MATERIAL DEFINITIONS 2.11
2.3 DAMAGE DEFINITIONS 2.11
2.3.1 Plate Damage *2.12*
2.3.2 Stiffener Damage *2.13*
2.4 DATABASES 2.14
2.4.1 Load-Shortening Curve Databases *2.14*
2.4.1.1 The NS94D LS-Curve Database 2.14
2.4.1.2 The VAST LS-Curve Database 2.15
2.4.1.3 The Special LS-Curve Database 2.16
2.4.1.4 The Project LS-Curve Database 2.17
2.4.2 Other Databases *2.17*
2.4.2.1 The Stiffener Database 2.17
2.4.2.2 The Material Database 2.18
2.4.2.3 The Damage Database 2.19
2.4.2.4 The SSCP23 Critical Collapse Stress Database 2.19
2.5 IMPORTANT ALGORITHMS 2.20
2.5.1 Importing a Cross-section from MGDSA or DIGFEM *2.20*
2.5.1.1 Introduction 2.20
2.5.1.2 The Algorithm 2.20

P515416.PDF [Page: 12 of 97]

2.5.1.3 Reading MGDSA Input Using Class MGDSAReader ... 2 22
2.5.2 Assigning Load-Shortening Curves ... *2 25*
2.5.2.1 Assigning of Load-Shortening Curves from the VAST and NS94D LS-Curve Databases ... 2.25
2.5.2.2 Generating Load-Shortening Curves for Longitudinally Stiffened Panels via FEA ... 2.26
2.5.3 Calculating SSCP23 Limit States ... *2.30*
2.5.3.1 Transversely Stiffened Panels ... 2 30
2.5.3.1.1 Yielding (Limit State 4) ... 2.30
2 5.3.1.2 Buckling (Limit State 1) ... 2 31
2.5.3.2 Longitudinally Stiffened Panels ... 2.31
2.5 3 2.1 Yielding (Limit State 4) ... 2 32
2.5.3.2 2 Buckling (Limit State 2) ... 2 32
2.5.3.2.3 Tripping (Limit State 3) ... 2.32
2.5.3.3 Hard Corners and User-Defined Structures ... 2 33
2.5.3.4 Calculating Component Stresses from Cross-section Bending Moments ... 2 33
2.5.4 Running an Ultimate Strength Analysis ... *2.34*
2.5.4.1 Generating Input Data for Ultimate Strength Analysis Programs ... 2 35
2.5.4.1.1 Generating Input for ULTMAT ... 2.35
2.5.4 1.2 Generating Input for NS94A ... 2.36
2.5.4.1.3 Generating Input for NS94B ... 2.37
2 5.4.1 4 Generating Input for NS94C ... 2 39
2.5 4.1.5 Generating Input for NS94D ... 2 40
2.5.4 1.6 Executing Ultimate Strength Analysis Programs ... 2 41
2.5.4.1.7 Creating Command Lines ... 2 41
2.5 4.1.8 Launching Ultimate Strength Analysis Programs ... 2 41
2 5.4.1 9 Reading Results from Ultimate Strength Analysis Programs ... 2 42
2.5.4.1 10 Reading Results From ULTMAT ... 2 42
2 5.4 1 11 Reading Results From NS94A ... 2 43
2 5.4.1.12 Reading Results From NS94B ... 2.43
2 5.4 1.13 Reading Results From NS94C ... 2.44
2.5.4.1.14 Reading Results From NS94D ... 2.44
2.5 4.1 15 Displaying Analysis Results ... 2 45

3 REFERENCES ... 3-1

APPENDIX A (ULTIMATE STRENGTH SYSTEM DIRECTORIES AND FILES) ... 1

APPENDIX B (PROJECT FILES) ... 5

Introduction ... 5
The Prefix ... 5
The USP File ... 5
The HOD File ... 5
The HDB File ... 6
The RLT File ... 6

APPENDIX C (MGDSA CROSS-SECTION FILE FORMAT) ... 7

APPENDIX D (FILE AVERAGE_YIELD_STRESS_TSP.TXT) ... 9

APPENDIX E (AVERAGE_YIELD_STRESS_LSP.TXT) ... 10

APPENDIX F (SERIALIZATION/FORMATS OF OBJECTS) ... 13

SFKeyPoint ... 13
SFPlate ... 13
SFStiffener ... 13
SFTransverselyStiffenedPanel ... 14
SFLongitudinallyStiffenedPanel ... 14
SFHardCorner ... 14

SFUserDefinedStructure 15
CrossSection 15

APPENDIX G (IMPORTANT CLASSES) 17

Introduction 17
CUSApp 17
CUSDoc 17
CUSView 17
HOODDocument 18
CrossSection 18
SFPlate 19
SFStiffener 19
SFLongitudinallyStiffenedPanel 19
SFTransverslyStiffenedPanel 19
SFHardCorner 20
SFUserDefinedStructure 20
MGDSAReader 20
NS94AWriter 20
NS94BWriter 21
NS94CWriter 21
NS94DWriter 21
ULTMATWriter 21
NS94AReader 21
NS94BReader 21
NS94CReader 22
NS94DReader 22
ULTMATReader 22
LSPanelWriter 22
VASTReader 22
FABSTRANLSCurveReader 22
ASASNLLSCurveReader 23
VASTDBLSCReader 23
NS94DLSCReader 23
SpecialLSCDBInterface 23
SSCP23RatioGenerator 23
NS94DCurveGenerator 24
LSCurveGenerator 24
GeneralLSCurveGenerator 24
Interpolator 24

APPENDIX H (NS94B INTERACTION CURVE RATIOS) 25

LIST OF FIGURES

FIGURE 2.1. TRANSVERSELY STIFFENED PANEL ANGLES 2.5
FIGURE 2.2. PLATES IN A TRANSVERSELY STIFFENED PANEL (BEFORE) 2.5
FIGURE 2.3. PLATES IN A TRANSVERSELY STIFFENED PANEL (AFTER) 2.6
FIGURE 2.4. LONGITUDINALLY STIFFENED PANEL ANGLES 2.7
FIGURE 2.5. PLATES IN A LONGITUDINALLY STIFFENED PANELS (BEFORE) 2.7
FIGURE 2.6. PLATES IN A LONGITUDINALLY STIFFENED PANEL (AFTER) 2.7
FIGURE 2.7. PLATES IN A HARD CORNER (BEFORE) 2.8
FIGURE 2.8. PLATES IN A HARD CORNER (AFTER) 2.9
FIGURE 2.9. HARD CORNER ANGLES 2.9
FIGURE 2.10. USER DEFINED STRUCTURE PLATES (BEFORE) 2.10
FIGURE 2.11. USER DEFINED STRUCTURE PLATES (AFTER) 2.10

LIST OF TABLES

TABLE 2.1. SSCP23 LIMIT STATES..... 2.30

1 Background

This document comprises the second part of the report [1] about the Ultimate Strength application and is concerned with its software aspects only.

This section provides background information about the Ultimate Strength system. Section 1.1 provides a general introduction to its functionality and Section 1.2 gives a very general overview of the system architecture.

1.1 Introduction

The purpose of the Ultimate Strength application is to provide the user with a straightforward, user friendly approach to analyzing the ultimate strength of a ship cross-section.

The primary source of cross-section input is MGDSA [2]. There is also more limited support for importing cross-sections from DIGFEM [3, 4].

The Ultimate Strength program is capable of generating two types of ultimate strength results: 1) moment-curvature and 2) bi-axial interaction. To generate these results the program makes use of ULTMAT and the NS94 suite of programs [5, 6] in conjunction with a set of load-shortening curves obtained from the VAST [7], NS94D, and a Special (User-Generated) LS-Curve databases. Only certain combinations of analysis type, ultimate strength program, and load-shortening curve databases are permitted.

When a user imports a cross-section from MGDSA or DIGFEM, the Ultimate Strength program analyzes its structure and identifies stiffened panels and hard corners and assigns each one a load-shortening curve that is interpolated from curves available in the VAST and/or NS94D databases.

The Ultimate Strength program provides two modes of interacting with a cross-section. The first mode, called the 'Finite Element' mode, represents the cross-section in terms of plates, stiffeners, and key-points. The second mode, called the 'Structural Units' mode, represents the cross-section in terms of stiffened panels and hard corners.

In the 'Finite Element' mode, the user has the option of modifying dimensions and material properties of plates and stiffeners. Given that project settings permit the use of the Special LS-Curve database, the user can also specify damage.

In the 'Structural Units' mode, the user has the option of disabling structural units and viewing their SSCP23 limit states [8, 9]. If project settings permit the use of the Special LS-Curve database, then the user can also assign to a unit an arbitrary load-shortening curve from a database or generate one using one of the FEA programs (VAST, FABSTRAN [10], ASAS-NL [11]).

Given that the MGDSA cross-section data contains applied horizontal and vertical bending moments the user can determine which units will fail by viewing the SSCP23 results.

1.2 System Architecture

Logically, the Ultimate Strength System is divided into nine major sub-systems:

1) The Input Module
2) The Material Database
3) The Damage Database
4) The Beam Database
5) The Load-Shortening Curve Database
6) The Ultimate Strength Analysis Module

7) The Finite Element Analysis Module
8) The HOOD [12] Core Module and
9) The Main Shell Module

1.2.1 The Input Sub-system

The purpose of the Input sub-system is to read definitions of cross-sections that are not in HOOD format and translate them into cross-sections that are.

The sub-system consists of two readers. One is capable of reading a cross-section definition from a MGDSA '.usd' file and the other is capable of reading a DIGFEM cross-section definition stored in '.tbs', '.tta', '.tmp', and '.mfl' files.

The translation process involves the creation of a basic model of geometry of a cross-section and populating the stiffener and material kind databases any time new stiffener or material kind definitions are encountered. The readers are not responsible for identifying structural units, assigning load-shortening curves, or performing any other calculations on the generated cross-section.

1.2.2 The Material Database

The purpose of the Material Database sub-system is to store and manage material definitions associated with plates and stiffeners that make up a cross-section.

This sub-system is composed of a database manager and a data file. The data file is used to store material definitions between program runs and may contain definitions from many different cross-sections. The database manager is responsible for storing material definitions at run time and for handling queries, insertions, and updates.

1.2.3 The Damage Database

The purpose of the Damage Database sub-system is to store and manage damage definitions associated with plates and stiffeners that make up a cross-section.

This sub-system is composed of a database manager and a data file. The data file is used to store damage definitions between program runs and may contain definitions from many different cross-sections. The database manager is responsible for storing damage definitions at run time and for handling queries, insertions, updates and deletions.

1.2.4 The Beam Database

The purpose of the Beam Database sub-system is to store and manage definitions of T-beams that may be used in defining a cross-section.

This sub-system is composed of a database manager and three data files. The data files are used to store beam definitions between program runs and may contain definitions from many different cross-sections. The beam database manager is responsible for storing beam definitions at run time and for handling queries, insertions, updates and deletions.

1.2.5 The Load-Shortening Curve Database

The Load-Shortening Curve Database sub-system is responsible for storing pre-generated (base) load-shortening curve definitions and for producing (via interpolation) load-shortening curves that are particular to stiffened panels for each given cross-section. The sub-system is also responsible for managing the curves that have been interpolated.

The Load-Shortening Curve Database sub-system consists of database files containing the base curve definitions and load-shortening curve readers and generators (interpolators).

The choice of curves that are used for interpolation depends on the selected load-shortening curve database and on the plate slenderness and column slenderness ratios for longitudinally stiffened panels, and plate slenderness and panel aspect ratios for transversely stiffened panels.

The two major types of LS-Curve databases that can be used for curve interpolation are the VAST database and the NS94D database.

The VAST database contains curves for longitudinally stiffened panels. These curves were generated using either the end-to-end model or the center to center model. For each boundary condition, the the VAST database is further subdivided into nine sets of sixteen boundary conditions, corresponding to diffener stiffener to total area ratios of 0.2, 0.275, 0.35 and maximum coefficients of plate imperfection of 0.1, 0.2, and 0.3. Plate slenderness ratios range from 1.0 to 4.0. Column slenderness ratios range from 0.3 to 1.2.

The NS94D database contains curves for longitudinally and transversely stiffened panels. For longitudinally stiffened panels, plate slenderness ratios range from 1.0 to 4.0 and column slenderness ratios range from 0.2 to 1.2. For transversely stiffened panels, plate slenderness ratios range from 1.0 to 4.0 and panel aspect ratios range from 1.0 to infinity.

Load-shortening curves for longitudinally stiffened panels are derived using bi-linear interpolation. Load-shortening curves for transversely stiffened panels are derived using linear-exponential interpolation (linear on plate slenderness ratios and exponential on aspect ratios).

In addition to databases used for interpolation, this sub-system contains a special load-shortening curve database containing curves that have been generated by the user with FEA programs like FABSTRAN.

1.2.6 The Ultimate Strength Analysis Sub-system

The purpose of the Ultimate Strength Analysis sub-system is to generate ultimate strength results such as moment-curvature curves and bi-axial interaction curves.

This sub-system consists of ultimate strength analysis programs ULTMAT, NS94A, NS94B, NS94C, and NS94D, and interface objects capable of translating cross-section definitions into formats particular to the ultimate strength programs and of translating results from the ultimate strength programs into format used in the Ultimate Strength application's data files.

1.2.7 The Finite Element Analysis Sub-system

The Finite Element Analysis sub-system provides the Ultimate Strength application with the ability to generate load-shortening curves for longitudinally stiffened panels, including those that have been damaged.

It consists of programs LSPanel [13], VAST, FABSTRAN, and ASAS-NL. The latter three programs are responsible for performing finite element analyses. LSPanel is responsible for generating FE analysis input.

The sub-system also contains a writer object that is responsible for translating HOOD definitions of longitudinally stiffened panels into a format understood by

LSPanel, and readers, which are capable of extracting load-shortening curve definitions from output files produced by FE analysis.

1.2.8 The HOOD Core Sub-system

The HOOD Core Module includes the basic HOOD framework and extends its functionality for the purposes of the Ultimate Strength application. It integrates with most of the other modules and is responsible for managing things like analysis options, databases, and indirectly user input.

1.2.9 The Main Shell Sub-system

The Main Shell sub-system comprises the system specific layer of the Ultimate Strength application. This module is responsible for providing and managing the Graphical User Interface (GUI), handling events and routing some to the HOOD Core sub-system, and performing system specific tasks that cannot be handled by the HOOD Core Module sub-system.

2 IMPLEMENTATION

This section provides a high level description of the implementation of the Ultimate Strength program. It covers topic like the representation of a cross-section, databases, and important algorithms that are not immediately apparent by looking at the source code.

2.1 The Representation of a Cross-section

Ultimate strength analyses are performed on a cross-section of a ship, which in the Ultimate Strength application is represented by a CrossSection object.

In its basic form, the geometry of a cross-section is represented as a set of basic structural elements, plates and stiffeners, which are connected at junctions that are referred to as key-points. Each of these elements is associated with a material and damage definition from the material type and damage type databases and the stiffeners are associated with beam definitions from the stiffener type database. It should be noted that there is no direct correspondence between these structural elements and the elements in FE models.

This basic form of describing a cross-section is insufficient for the purposes of performing ultimate strength analyses. Consequently, each cross-section has a second level of abstraction where the plates and stiffeners are used to form structural units like stiffened panels, hard corners, and user-defined structures.

The two modes of representing cross-sections afford the user with different levels of interaction.

In the 'Finite Element' mode, the user can change dimensions and material properties of plates and stiffeners. If the ultimate strength analysis program selected by

the user allows the use of user-defined load-shortening curves, then the user can also define plate and stiffener damage.

In the 'Structural Units' mode, the user can enable or disable structural units, like hard corners and stiffened panels, and view their load-shortening curves, dimensions, material properties, damage definitions, and SSCP23 limit states. If the ultimate strength analysis program selected by the user allows the use of user-defined load-shortening curves, then the user can also assign an arbitrary load-shortening curve from the database or generate a new curve using one of the FEA programs.

The following sections describe the building blocks of a cross-section. Please see the source code for implementation details.

2.1.1 Geometric Elements

There are seven classes of elements that comprise cross-sectional geometry:

1) key-points
2) plates
3) stiffeners
4) transversely stiffened panels
5) longitudinally stiffened panels
6) hard corners
7) user-defined structures

2.1.1.1 Key-Points

A key-point is a point in 2D space that is used to form plate-plate and plate-stiffener junctions. For most ultimate strength analysis programs like NS94A, key-

points are used to indicate the locations of structural units such as stiffened panels and hard corners.

In the Ultimate Strength application, key-points are represented using objects of class SFKeyPoint. For more information on class SFKeyPoint please see the internal Ultimate Strength documentation.

2.1.1.2 Basic Structural Elements

The basic structural elements of a cross-section are plates and stiffeners, which are described in this section.

2.1.1.2.1 Plates

A plate is a three dimensional object. In the Ultimate Strength application, it is represented as a line segment. The line segment is terminated by key-points, which define the plate width. A plate also has a length and a thickness and is associated with a material definition and a damage definition.

In the Ultimate Strength application, plates are represented using objects of class SFPlate. For more information on class SFPlate please see the internal Ultimate Strength documentation.

2.1.1.2.2 Stiffeners

A stiffener is a three dimensional object. All stiffeners supported by the Ultimate Strength application are T-beams, which are represented in terms of two line-segments. The base of a stiffener is attached to a key-point and its orientation is determined by a direction vector. The dimensions of a stiffener cross-section are defined by a record in the stiffener type (beam) database. The length of a stiffener is

stored within the stiffener object. A stiffener is also associated with a material definition and a damage definition.

In the Ultimate Strength application, stiffeners are represented using objects of class SFStiffener. For more information on class SFStiffener please see the internal Ultimate Strength documentation.

2.1.1.3 Structural Units

Ultimate strength analysis programs like ULTMAT cannot accept a cross-section that is defined in terms of basic structural elements like plates and stiffeners. Rather, they need the cross-section definition to be in terms of structural units like stiffened panels and hard corners, which can be assigned load-shortening curves. To achieve this, CrossSection objects are able to translate a representation that is in terms of basic structural elements into one that is in terms of structural units. The structural units that can be created are described in this section.

With the exception of user-defined structures, each structural unit contains a single implicit key-point that is the junction of all the plates and stiffeners that comprise the unit. User-defined structures, which are essentially a network of structural units, may contain multiple implicit key-points.

Since plates are associated with two key-points, it is possible that a single plate object comprises two separate structural units. Therefore, only the half of the plate whose end-point is the key-point included in the structural unit is considered to be part of that structural unit. Stiffeners, which are associated with only one key-point are simply considered to be part of the structural unit that includes that key-point.

2.1.1.3.1 Transversely Stiffened Panels

A transversely stiffened panel is defined in terms of two plates that share a common key-point and meet at a minimum angle of 150 degrees (Fig. 2.1). To form a panel, plate halves that are closest to the common key-point are used (Fig. 2.2 and Fig. 2.3). The length, material and damage properties of a transversely stiffened panel are based on the combination of the properties of its constituent plates. These properties are combined using a weighted average based on the cross-sectional area of each basic structural element. As a structural unit, a transversely stiffened panel may be associated with a load-shortening curve. Note that like any structural unit, a transversely stiffened panel may be re-designated as a user-defined structure.

In the Ultimate Strength application, transversely stiffened panels are represented using objects of class SFTransverselyStiffenedPanel. For more information on class SFTransverselyStiffenedPanel please see the internal Ultimate Strength documentation.

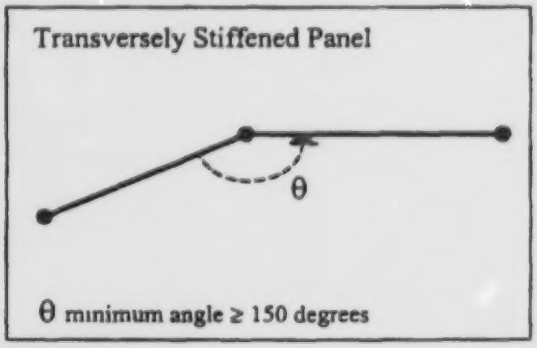


Figure 2.1. Transversely Stiffened Panel Angles

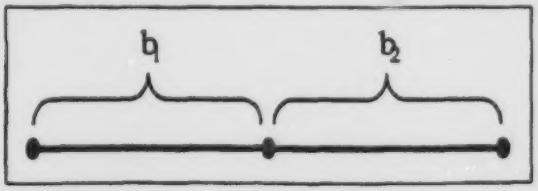


Figure 2.2. Plates in a Transversely Stiffened Panel (Before)

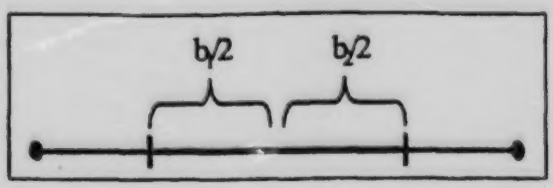


Figure 2.3. Plates in a Transversely Stiffened Panel (After)

2.1.1.3.2 Longitudinally Stiffened Panels

A longitudinally stiffened panel is defined in terms of two plates and a stiffener that share a common key-point. The points must meet at an minimum angle of 150 degrees (Fig. 2.4). To form a panel, plate halves that are closest to the common key-point are used (Fig. 2.5 and Fig. 2.6). The material and damage properties of a longitudinally stiffened panel are based on the combination of the properties of its constituent plates and stiffener. These properties are combined using a weighted average based on the cross-sectional area of each basic structural element. As a structural unit, a longitudinally stiffened panel may be associated with a load-shortening curve. Note that like any structural unit, a longitudinally stiffened panel may be re-designated as a user-defined structure.

In the Ultimate Strength application, longitudinally stiffened panels are represented using objects of class SFLongitudinallyStiffenedPanel. For more information on class SFLongitudinallyStiffenedPanel please see the internal Ultimate Strength documentation.

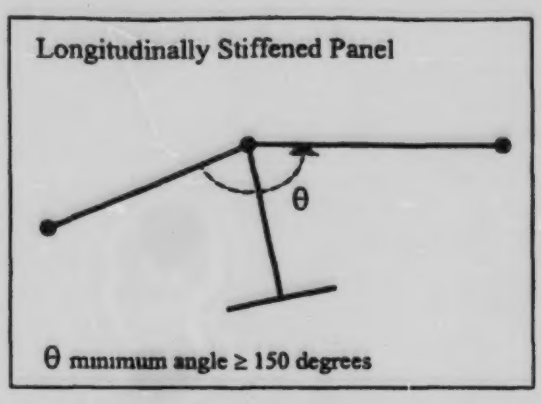


Figure 2.4. Longitudinally Stiffened Panel Angles

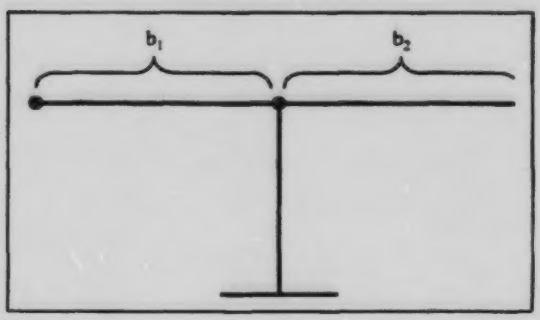


Figure 2.5. Plates in a Longitudinally Stiffened Panels (Before)

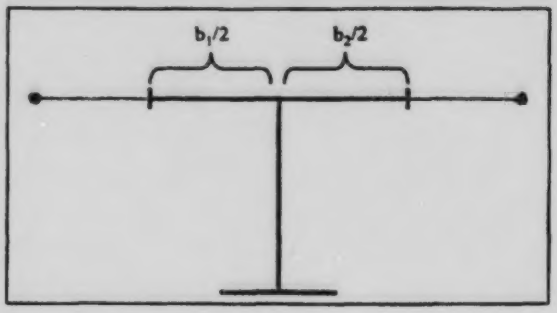


Figure 2.6. Plates in a Longitudinally Stiffened Panel (After)

2.1.1.3.3 Hard Corners

A hard corner is any combination of plates and stiffeners that does not match the definition of a transversely or longitudinally stiffened panel. To form a hard corner, plate halves that are closest to the common key-point are used (Fig. 2.7 and Fig. 2.8). Thus two plates meeting at a minimum angle of less then 150 degrees (Fig. 2.9), or three or more plates sharing a key-point will constitute a hard corner. A hard corner may contain stiffeners. The length, material and damage properties of a hard corner are based on the combination of the properties of its constituent plates and stiffeners. These properties are combined using a weighted average based on the cross-sectional area of each basic structural element. As a structural unit, a hard corner may be associated with a load-shortening curve. Note that like any structural unit, a hard corner may be re-designated as a user-defined structure.

In the Ultimate Strength application, hard corners are represented using objects of class SFHardCorner. For more information on class SFHardCorner please see the internal Ultimate Strength documentation.

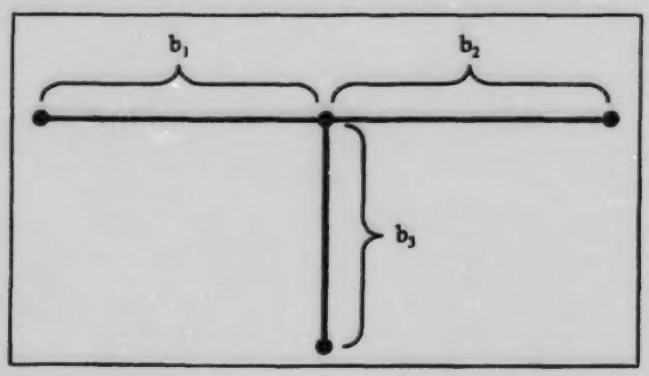


Figure 2.7. Plates in a Hard Corner (Before)

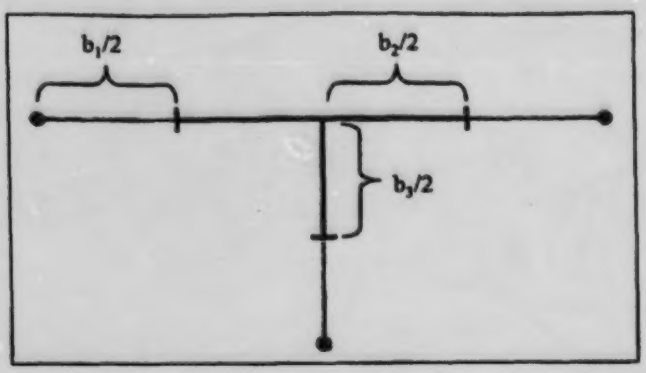


Figure 2.8. Plates in a Hard Corner (After)

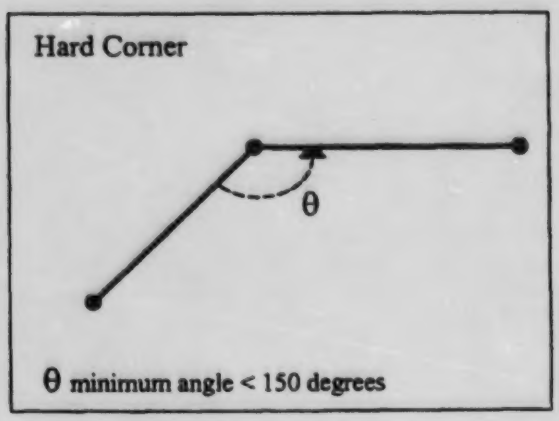


Figure 2.9. Hard Corner Angles

2.1.1.3.4 User-Defined Structures

A user-defined structure is a combination of one or more stiffened panels and hard corners. All structural units that constitute a user-defined structure must form network. Figures 2.10 and 2.11 show an example of a user defined structure in the plate-stiffener and stiffened panels-hard corner modes, respectively.

The length, material and damage properties of a user-defined structure are based on the combination of the properties of its constituent plates and stiffeners. These

properties are combined using a weighted average based on the cross-sectional area of each basic structural element. As a structural unit, a user-defined structure may be associated with a load-shortening curve.

In the Ultimate Strength application, user-defined structures are represented using objects of class SFUserDefinedStructure. For more information on class SFUserDefinedStructure please see the internal Ultimate Strength documentation.

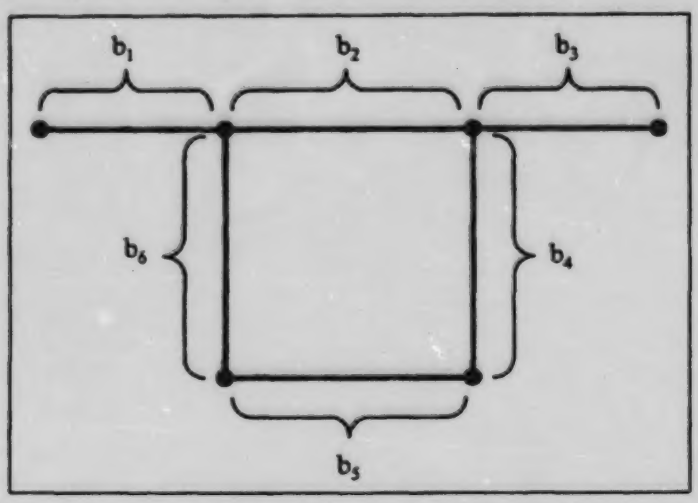


Figure 2.10. User Defined Structure Plates (Before)

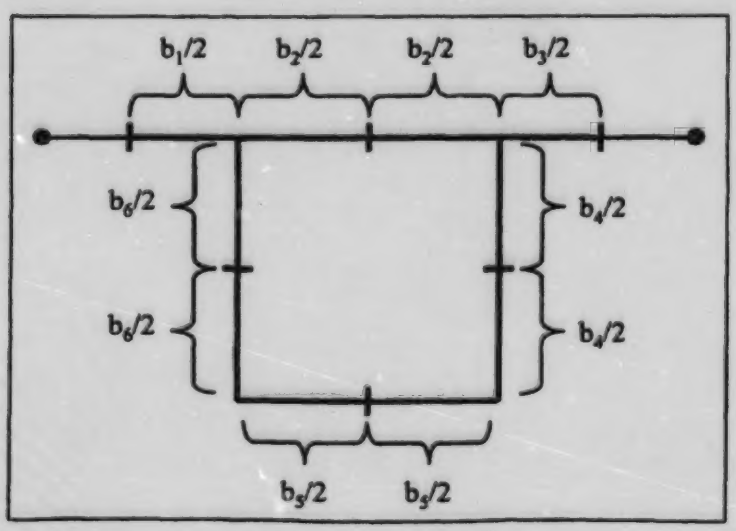


Figure 2.11. User Defined Structure Plates (After)

2.2 Material Definitions

Material definitions of structural units, like stiffened panels, are derived from material definitions of their constituent plates and stiffeners, as described in the previous section. The definitions associated with plates and stiffeners are represented by instances of class MaterialKind and are stored in a database. They include the modulus of elasticity, Poisson's ratio and yield stress.

2.3 Damage Definitions

Like material definitions, damage definitions of structural units are based on damage definitions of their constituent plates and stiffeners. The definitions associated with plates and stiffeners are represented by instances of classes PlateDamage and StiffenerDamage, respectively, and are stored in a database. They include information about distortion and corrosion.

It should be noted that the user has the option of defining new damage but does not have to, as the Ultimate Strength program assigns default damage definitions to each plate and stiffener when a new cross-section is created.

It should also be noted that while the Ultimate Strength application has facilities for modeling damage, damage information is useful only if the load-shortening curve of a specified stiffened panel needs to be calculated using a non-linear FEA program. An option to calculate such a curve is available only when:

1) The selected ultimate strength program can handle load-shortening curves stored in the Special LS-Curve database. (Only NS94A and NS94B support this feature.)

2) The damage is applied to a longitudinally stiffened panel, as this is the only structural unit for which LSPanel can generate FEA input, and thus the only unit for which a load-shortening curve can be produced automatically.

2.3.1 Plate Damage

Plate damage definitions are represented by instances of class PlateDamage and contain information about distortion and corrosion.

Distortion is applied at a plate's center. The value that is specified by the user represents the amplitude of the half-sine waveform that models the shape of the distorted plate. The pattern of the sine wave distortions depends on the form of imperfection, as discussed in the LSPanel documentation.

In the damage database, plate distortions are stored in terms of coefficients of plate imperfection. The default value for the coefficient of plate imperfection is 0.15, which represents an average imperfection. The actual imperfection (distortion) is computed by multiplying the coefficient of imperfection by the plate thickness and the square of the plate slenderness ratio.

Plate corrosion is defined in terms of a combination of three basic corrosion patters: 1) uniform, 2) sinusoidal, and 3) edge reductions. The value that is specified by the user for uniform corrosion represents the amount by which the plate thickness has been reduced across the entire plate area. The value that is specified by the user for sinusoidal corrosion represents the amount by which the plate thickness has been reduced at the center (i.e. it is the amplitude of the half-sine waveform that models the corrosion). Values that are specified for edge corrosion represent the amount by which the plate thickness has been reduced at each particular edge. Default values for all the corrosion parameters are 0, indicating no corrosion.

An in-depth description of these corrosion patterns and their combinations are provided in section 3.1 of *Incorporation of UK NS94 Ultimate Strength Analysis in MGDSA Part I: Engineering Considerations*.

2.3.2 Stiffener Damage

Stiffener damage definitions are represented by instances of class StiffenerDamage and contain information about distortion and corrosion.

A distortion is applied at a stiffener's center. The value that is specified by the user represents the amplitude of the half-sine waveform that models the shape of the distorted stiffener.

In the damage database, stiffener distortions are stored in terms of coefficients of stiffener imperfection. The default value for the coefficient of stiffener imperfection is 0.002. The actual imperfection (distortion) is computed by multiplying the coefficient of the stiffener imperfection by the stiffener length.

Stiffener corrosion is defined separately for the stiffener flange and the stiffener web, but both are defined in terms of a combination of three basic corrosion patterns: 1) uniform, 2) sinusoidal, and 3) edge reduction.

An in-depth description of these corrosion patterns and their combinations are provided in section 3.1 of *Incorporation of UK NS94 Ultimate Strength Analysis in MGDSA Part I: Engineering Considerations*.

2.4 Databases

The Ultimate Strength program makes use of a number of databases, which can be divided into databases containing load-shortening curve information and databases containing other types of information such as material and damage definitions.

2.4.1 Load-Shortening Curve Databases

There are four load-shortening curve databases, all of which contain curves that are in a normalized form. These four databases are:

1) The NS94D LS-Curve Database
2) The VAST LS-Curve Database
3) The Special LS-Curve Database
4) The Project LS-Curve Database

NS94D and VAST LS-Curve databases contain load-shortening curves that are defined for a specific set of parameter values (α and β, or λ and β) and are used to generated new curves through interpolation. The Special LS-Curve database contains curves that were generated and imported by the user. The Project LS-Curve database contains curves specific to a particular cross-section and may contain a mixture of curves interpolated from the NS94D and VAST LS-Curve databases and curves from the Special LS-Curve database.

2.4.1.1 The NS94D LS-Curve Database

The NS94D LS-Curve database contains curves for longitudinally stiffened panels with plate slenderness ratios ranging from 1.0 to 4.0 and column slenderness ratios ranging from 0.2 and 1.2, and transversely stiffened panels with plate slenderness ratios ranging from 1.0 to 4.0 and panel aspect ratios ranging from 1.0 to infinity.

The database is made up of two files, which are stored in the 'US LSC DBs\NS94D DB' directory. File 'NS94D_LCurves.dat' contains twenty four curve definitions for longitudinally stiffened panels. File 'NS94D_TCurves.dat' contains eight curve definitions for transversely stiffened panels.

In the Ultimate Strength application, load-shortening curves stored in the NS94D database are used in generating curves for specific values of plate and column slenderness ratios, and plate and aspect ratios, using interpolation.

The low level accessing of the NS94D database and interpolation take place using an instance of class NS94DLSCurveGenerator, which is an attribute of the CUSApp object. Curves for longitudinally stiffened panels are generated using bi-linear interpolation and curves for transversely stiffened panels are generated using a linear-exponential interpolation (linear on plate slenderness ratios and exponential on aspect ratios as in the NS94D program).

2.4.1.2 The VAST LS-Curve Database

The VAST LS-Curve database contains curves for longitudinally stiffened panels.

The database is stored in a directory called 'US LSC DBs\VAST DB', which is subdivided into eighteen subdirectories. The subdirectories' names are made up of three alphanumeric characters. The first character is either the letter 'c' or the letter 'e', indicating that the directory contains curves for the center-to-center model and the end-to-end model, respectively. The second character is either a 1, 2, or 3, indicating that the directory contains curves for stiffener area to total stiffened panel area ratios of 0.2, 0.275, and 0.35, respectively. The last character is either a 1, 2, or 3, indicating that

the directory contains curves for maximum plate coefficient of imperfection of 0.1, 0.2, and 0.3.

The prefix of each file contains five characters. The first character of the file name is the same as the first character of the directory name. The second character is either a 1, 2, 3, or 4, and indicates whether the file contains a curve for a plate slenderness ratio of 1.0, 2.0, 3.0, or 4.0. The third character is also a 1, 2, 3, or 4, and indicates whether the file contains a curve for a column slenderness ratio of 0.3, 0.6, 0.9, or 1.2. The last two characters are the same as the last two characters of the directory name.

In the Ultimate Strength application, load-shortening curves stored in the VAST database are used in generating curves for specific values of plate and column slenderness ratios using bi-linear interpolation. The low level accessing of the VAST database and interpolation take place in an instance of class LSCurveGenerator, which is an attribute of the CUSApp object.

2.4.1.3 The Special LS-Curve Database

The Special LS-Curve Database contains curves generated by the user. These curves must be normalized, but they may represent any type of structure.

The database is stored in a directory called 'US LSC DBs/SPECIAL DB'. The database is comprised of a master file, which contains a list of file name-curve name pairs, and the files that contain the actual load-shortening curve definitions.

The filename of each file with a load-shortening curve definition consists of a prefix that contains the string 'LSC' followed by eight characters representing a number from 0 to 99,999,999, followed by an extension '.LSC'.

The low level accessing of the Special LS-Curve database takes place using an instance of class SpecialLSCDBInterface, which is an attribute of the CUSApp object.

2.4.1.4 The Project LS-Curve Database

The LS-Curve database associated with a project contains load-shortening curves that have been generated specifically for the cross-section defined in that project, including curves provided by the user.

The database is stored in the project directory with the project file. It shares the same prefix as the project file, but its extension is '.HDB'.

2.4.2 Other Databases

In addition to load-shortening curve databases, the Ultimate Strength application makes use of four other databases:

1) The Stiffener Database
2) The Material Database
3) The Damage Database
4) The SSCP23 Average Yield Stress Database

2.4.2.1 The Stiffener Database

The Stiffener Database contains definitions of stiffeners that may be used in a cross-section.

The database is provided in three files located in the 'US HOOD DBs' directory. File 'HOOD_BEAM_DB.HDB' is the root file, with a WWWInLine import of files 'HOOD_BEAM_DB_METRIC.HDB' and

'HOOD_BEAM_DB_IMPERIAL.HDB', which contain beams defined in metric units (millimeters) and imperial units, respectively.

Since at the moment input data to the Ultimate Strength application does not provide information regarding the units of measurements, the application assumes that beam definitions use the metric system.

When the Ultimate Strength application is launched, the definitions in the database files are loaded into objects of class TBeam, which are placed in an instance of class StiffenerKindDB responsible for their management. (The StiffenerKindDB object is an attribute of the CUSApp (application) object.) When the Ultimate Strength application is terminated, the contents of the in-memory database are saved back into the data files.

2.4.2.2 The Material Database

The Material Database contains definitions of materials that may be assigned to plates and stiffeners.

The file that contains the database is called 'HOOD_MATERIAL_DB.HDB' and is found in the 'US HOOD DBs' directory. When the Ultimate Strength application is launched, the definitions in this file are loaded into objects of class MaterialKind, which are placed in an instance of class MaterialKindDB responsible for their management. (The MaterialKindDB object is an attribute of the CUSApp (application) object.) When the Ultimate Strength application is terminated, the contents of the in-memory database are saved back into the data file.

2.4.2.3 The Damage Database

The Damage Database contains definitions of damage that may be assigned to plates and stiffeners.

The file that contains the database is called 'HOOD_DAMAGE_DB.HDB' and is found in the 'US HOOD DBs' directory. When the Ultimate Strength application is launched, the definitions in this file are loaded into objects of class DamagePlate, for plate damage definitions, and objects of class DamageStiffener, for stiffener damage definitions. The DamagePlate and DamageStiffener objects are placed in an instance of class DamageKindDB, which is responsible for their management. (The DamageKindDB object is an attribute of the CUSApp (application) object.) When the Ultimate Strength application is terminated the contents of the in-memory database are saved back into the data file.

2.4.2.4 The SSCP23 Critical Collapse Stress Database

The SSCP23 Critical Collapse Stress Database contains definitions of critical collapse stress to yield stress ratios for transversely and longitudinally stiffened panels, which are use in calculations of SSCP23 limit states 1 and 2 (buckling of plating between stiffeners and buckling of longitudinally stiffened panel).

The database consists of two files located in the 'Sscp23' directory. The file 'Average_Yield_Stress_LSP.txt' contains the ratios for longitudinally stiffened panels. The file 'Average_Yield_Stress_TSP.txt' contains the ratios for transversely stiffened panels.

The interface to the data files is provided via an instance of class SSCP23RatioGenerator, which is an attribute of the CUSApp object. Unlike the other databases in this section, the contents of the data files are never loaded into memory.

Instead the files are accessed each time the critical collapse stress must be calculated, which happens when stiffened panel dimensions or material properties change.

2.5 Important Algorithms

This section contains some of the more important algorithms used in the Ultimate Strength application.

2.5.1 Importing a Cross-section from MGDSA or DIGFEM

2.5.1.1 Introduction

Internally, the Ultimate Strength application manipulates cross-sections, which are represented in HOOD format. As it has no modeling capability, it imports cross-sections that are defined in MGDSA and DIGFEM formats. This section provides an overview of the algorithms used for importing such cross-sections.

The MGDSA file format is described in Appendix C. Section 2.2 provides a more detailed discussion of the algorithm used when reading MGDSA cross-section definitions.

2.5.1.2 The Algorithm

The first step in creating a new CrossSection object for use in the Ultimate Strength application is to read the definition of a cross-section stored in MGDSA file or DIGFEM files. This is achieved through the use of instances of class MGDSAReader and class DIGFEMReader, respectively.

The reading process imports material and stiffener definitions into the material kind and stiffener kind databases and creates a partial cross-section definition. This

definition specifies the cross-section geometry in terms of key-points, plates, and stiffeners. There are no structural units. In addition, the associations between key-points, plates, and stiffeners are in terms of identifiers stored in data files rather than actual pointer references. This also applies to associations between plates and stiffeners and records in the material kind and stiffener kind databases.

The CrossSection object produced by the reading process is then inserted into a newly created scene graph.

The next step involves linking of the CrossSection object to the material kind, stiffener kind, and damage databases, and the SSCP23 data generator.

When the databases have been linked with the CrossSection object, the import process sets up pointer references between plate, stiffener, and key-point objects based on IDs extracted during reading. IDs are also used to set up pointer references between plates and stiffeners and material definitions in the material kind database, and stiffeners and stiffener definitions in the stiffener kind database.

Once the basic geometry is ready, the process analyzes the cross-section and identifies and creates structural units like stiffened panels and hard corners.

Next each plate is assigned the default plate damage definition and each stiffener is assigned the default stiffener damage definition, which are always found in the damage database.

Structural units are assigned yielding stress ratios for SSCP23 Limit State calculations and load-shortening curves for ultimate strength analysis.

Those units that cannot be matched with a load-shortening curve are disabled.

If the original MGDSA or DIGFEM input contained applied bending moments, then their values are used to set stresses for each structural unit.

When this last step is done the CrossSection object is fully defined and ready to use.

Section 2.5.2.1 *Assigning of Load-Shortening Curves from the VAST and NS94D LS-Curve Databases* contains a description of the algorithm used for generating load-shortening curves. SSCP23 limit states are discussed in Section 2.5.3 *Calculating SSCP23 Limit States*.

2.5.1.3 Reading MGDSA Input Using Class MGDSAReader

MGDSA is the primary source of cross-sections for the Ultimate Strength application. MGDSA generated cross-section definitions are stored in files with the extension '.usd'. The file format for these files is described in Appendix A.

The responsibility for reading MGDSA input and translating it into an instance of class CrossSection falls to an instance of class MGDSAReader.

In the MGDSA file, plate definitions correspond directly to plate elements found in the global FE model. Stiffener locations, as provided, reflect the distribution of stiffeners along FE plate elements and not physical stiffener locations.

Key-points that define the base of each stiffener's web, do not necessarily have the same coordinates as the key-points that define plate end-points, and therefore web key-points are initially treated as different from the plate key-points.

By the definition of class CrossSection, each key-point must be an end-point of at least one plate. To satisfy this definition, the reader places each key-point that defines the base of a web onto the nearest plate. Some of the web key-points fall onto

or very near to key-points that end plates. In such cases, the plate key-point becomes the key-point of the web and the other key-point is discarded. (To determine what is 'close enough' with respect to such cases, a tolerance of 5% of the shortest FE plate element is used.) Those key-points that are not close enough to a plate end-point are simply moved onto the plate, effectively breaking the plate in two.

Sometimes, no web key-point is ever close enough to a plate end-point. When two plates share such a key-point and the minimum separation angle is not less than 150 degrees, the resulting section fits the definition of a transversely stiffened panel. During testing using the CPF model, it was discovered that usually many such sections were produced. This was deemed undesirable. Consequently, the algorithm was modified to remove key-points with no associated stiffeners, if they are associated with two plates, each having its other key-point associated with a stiffener. When such a key-point is removed the two plates and their thickness and material properties are merged using a weighted average based on the cross-sectional area of each original plate.

Testing has also revealed that some cross-sections of the CPF model contain plating definitions such that plates intersect without being attached. This condition appears to have no significant adverse effect on the Ultimate Strength program. Moreover, such cases occur mostly when the cross-section includes the superstructure, which bears little load and usually should not be included in an ultimate strength analysis. Consequently, no modifications were made to the reader algorithm to correct this inconsistency.

It was also discovered that the test CPF cross-section contains flat-bar beam definitions that mask as T-beam definitions. Basically, the flat-bar is modeled as a T without a flange. As a result, procedures used by the Ultimate Strength application for testing distance between points and lines, used when the user tries to select an object, would fail and crash it. To circumvent this problem, it was decided to modify the flat-bar definition slightly by creating a very small flange. This should have no significant

effect on moment calculations, but in the future a cleaner solution should be considered. The best option would be to provide full support for dealing with flat-bar beams throughout the Ultimate Strength application.

In addition to creating and associating cross-section geometry, the reader extracts definitions of material kinds and beams, and if it encounters any new ones, it places them in the material kind and stiffener kind databases, respectively.

The CrossSection object produced by the MGDSA reader contains information about:

1) the frame number,
2) the cross-section's coordinate along the axis running through the length of the global ship model,
3) the units of length,
4) the units of force,
5) the position of the structure with respect to the eye,
6) the information about how plates and stiffeners relate to key-points in terms of IDs but not actual object associations,
7) the information about which materials are used by each plate and stiffener in terms of IDs but not actual object associations,
8) the information about which beam definition is used by each stiffener in terms of IDs but not actual associations, and
9) the applied horizontal and vertical bending moments, if they were provided.

2.5.2 Assigning Load-Shortening Curves

2.5.2.1 Assigning of Load-Shortening Curves from the VAST and NS94D LS-Curve Databases

Load-shortening curves are assigned automatically to all structural units when a cross-section is created or upon request from the user.

All hard corners use a default load-shortening curve definition. Load-shortening curves that are assigned to stiffened panels are interpolated using curves found in the selected LS-Curve database. NS94D database contains definitions of curves for transversely and longitudinally stiffened panels. VAST database contains curves for longitudinally stiffened panels only, and if selected, the NS94D database is used to generate curves for transversely stiffened panels.

Curves for longitudinally stiffened panels are generated using bi-linear interpolation based on the plate and column slenderness ratios. Curves for transversely stiffened panels are generated using linear-exponential interpolation based on the column slenderness and aspect ratios. If any of the ratio values fall outside of the range supported by the selected database, then the values are snapped to the closest value in range. This introduces the possibility that a curve automatically generated for a stiffened panel is a poor approximation. In such cases, and if the current project options support it, the user has the option of assigning a different, better curve from the Special LS-Curve database.

Each load-shortening curve is generated by using an instance of class GeneralLSCurveGenerator, which serves as an interface to an instance of class NS94DCurveGenerator and an instance of class LSCurveGenerator. These two objects in turn provide an interface to the NS94D and VAST LS-Curve databases, respectively,

and they contain interpolators that are capable of generating load-shortening curves for a specific set of parameters.

In addition to the two generator objects, the GeneralLSCurveGenerator object contains a reference to a project database with load-shortening curves particular to a cross-section. If the parameters specified by the caller are for a curve that has already been generated and stored in the project database, a reference to that curve is returned. Otherwise the curve is generated by using the NS94DCurveGenerator or LSCurveGenerator objects, placed in the cross-section's load-shortening curve database, and its reference returned to the caller.

2.5.2.2 Generating Load-Shortening Curves for Longitudinally Stiffened Panels via FEA

Analysis programs NS94A and NS94B allow the user to specify any set of load shortening curves. To leverage this capability, the Ultimate Strength application gives the user the option to semi-automatically generate load-shortening curves for longitudinally stiffened panels.

The process of generating a load-shortening curve begins when the user presses the 'Generate Curve' button on the 'LS Curve' page of the longitudinally stiffened panel property sheet. The generated message invokes the method CLSCurvePage::OnGenerateBtn(), which passes the message onto the CUSView object, which informs the CUSDoc object, which calls the OnGenerateLSC() method.

The CUSDoc::OnGenerateLSC() method creates an instance of CLSPanelWizSheet and displays the dialog to the user. Through this dialog, the user can choose the FEA program (VAST, FABSTRAN, or ASAS-NL) and set global control parameters, boundary conditions, and meshing parameters that will be used by LSPanel to generate FEA input.

Once the analysis program and parameters have been selected, the OnGenerateLSC method creates an instance of class LSPanelWriter, sets its attributes based on the program and parameter choices and calls its write() method to generate an input file for LSPanel. The file is placed in the \<DIR>\US Temporary Files' directory, where <DIR> represents the directory of the UltimateStrength.EXE. The format of the LSPanel input file is described in Appendix A of *Incorporation of UK NS94 Ultimate Strength Analysis in MGDSA Part I: Engineering Considerations*.

After the file has been created, the method runs LSPanel, which generates input for the selected FEA program.

Next, it creates an instance of CLSPanelLSCurveDlg. This object is a dialog that allows the user to view and edit FEA input, run and re-run analyses, and preview and accept or reject resultant load-shortening curves.

Each FEA program is launched slightly differently.

When using VAST, input files are placed in the '\<DIR>\US Temporary Files' directory, where <DIR> represents the directory of the UltimateStrength.EXE. The full path to the directory together with the input files' prefix are passed to VAST.EXE as a command line parameter, when the VAST process is launched.

Once the analysis process is running, the method sets up a timer that is fired every second. Each time it triggers, it invokes the method OnTimer(), which checks whether the analysis process has finished. If it has, then the OnTimer() method reads the results, which are found in files <PREFIX>.T52 and <PREFIX>.GOM located in '\<DIR>\US Temporary Files' directory, where <PREFIX> represents the project file prefix and <DIR> represents the directory of the UltimateStrength.EXE. These files are read using an instance of class VASTLSCurveReader and the resultant

load-shortening curve is then plotted so that the user can either accept it, reject it, or modify the input file and re-run the analysis.

If the user specifies a name for the generated load-shortening curve and accepts it, then the dialog closes and the curve is inserted into the Special LSC-Curve database and assigned to the associated SFLongitudinallyStiffenedPanel object. Otherwise the dialog simply closes and no changes are made.

When using FABSTRAN, the 'LSPCurve.dat' input file is placed in 'C:\USTemp' directory. This directory is also used to store a 'pname.txt' file, which contains the path to the input file and the 'fabstran.bat' file, which when executed launches FABSTRAN.EXE. The contents of 'fabstran.bat' are two lines as follows:

```
@echo off
"D:\Program Files\Martec\Ultimate Strength\US Programs\FABSTRAN.EXE"<C:\USTemp\pname.txt>nul
```

The actual path to FABSTRAN.EXE may be different.

The reason for using this approach rather than the one taken with VAST is that FABSTRAN was not modified to handle Windows 95 style command lines, which may be long and contain white space. This approach ensures that all paths provided to FABSTRAN are valid and that it operates in a predictable working directory.

The process of running FABSTRAN is similar to that of VAST and uses the timer mechanism that polls for termination.

Once the analysis has finished, the OnTimer() method creates and instance of FABSTRANLSCurveReader and uses it to read the results from an output file whose path is 'C:\USTemp\LSPCurve.out'. The resultant load-shortening curve is normalized using the particular stiffened panel's area, length, yield stress, and modulus of

elasticity, and then displayed to the user so that the user can either accept it, reject it, or modify the LSPanel input file and re-run the analysis.

If the user specifies a name for the generated load-shortening curve and accepts it, then, like with VAST, the dialog closes and the curve is inserted into the Special LSC-Curve database and assigned to the associated SFLongitudinallyStiffenedPanel object. Otherwise the dialog simply closes and no changes are made.

With ASAS-NL it is necessary to run two executables. The first is PRENL.EXE and the second ASASNL.EXE.

The command line for launching PRENL.EXE is generated automatically and includes /DATA and /OUT parameters, which specify the paths and prefixes of the input and output files, respectively. The paths for all files are set to '\<DIR>\US Temporary Files' directory, where <DIR> represents the directory of the UltimateStrength.EXE. File prefixes are the same as the prefix of the Ultimate Strength project file. Once the PRENL process is running the dialog is closed. The user is responsible for running the second part of the analysis using ASASNL.EXE and for importing the load-shortening curve manually using the File:Import:Load Shortening Curve:ASAS-NL menu option and then assigning it to the desired longitudinally stiffened panel.

Running of an ASAS-NL analysis and testing of the various components and interfaces could not be completed as the programs and their data files were not available.

2.5.3 Calculating SSCP23 Limit States

The set of SSCP23 limit states that are supported by a particular structural unit depends on the type of that unit. Table 2.1 shows the SSCP23 limit states supported by each type of a structural unit.

Structural Unit Type	Supported SSCP23 limit states
Transversely Stiffened Panel	Yielding (Limit State 4), Buckling (Limit State 1)
Longitudinally Stiffened Panel	Yielding (Limit State 4), Buckling (Limit State 2), Tripping (Limit State 3)
Hard Corner	Yielding
User Defined Structure	Yielding

Table 2.1. SSCP23 Limit States

The following sections describe briefly the approach used in calculating SSCP23 limit states in the Ultimate Strength application. A more detailed discussion of SSCP23 limit states can be found in section 4 of *Incorporation of UK NS94 Ultimate Strength Analysis in MGDSA Part I: Engineering Considerations.*

2.5.3.1 Transversely Stiffened Panels

Limits states attributed to transversely stiffened panels are limit state 4 (yielding), and limit state 1 (buckling).

2.5.3.1.1 Yielding (Limit State 4)

The yielding limit state of a transversely stiffened panel is defined as the yield stress, which is a material property. Like the other material properties in the Ultimate Strength program, yield stress of a transversely stiffened panel is calculated as a weighted average of yield stress values of its component plates. The weight of each component is based on its cross-sectional area.

2.5.3.1.2 Buckling (Limit State 1)

The buckling limit of plating between stiffeners is calculated by multiplying the ratio of the critical collapse stress to yield stress by the yield stress of the transversely stiffened panel.

The yield stress is calculated as a weighted average of yield stress values of its component plates. The weight of each component is based on its area.

The ratio of critical collapse stress to yield stress is set by the CrossSection object when a SFTransverselyStiffenedPanel is created and by the HOODDocument object each time a SFTransverselyStiffenedPanel object is modified. The ratio is generated by an instance of class SSCP23Generator, which uses the value of the plate slenderness ratio of the transversely stiffened panel in question to calculate it by linearly interpolating data found in the file 'Average_Yield_Stress_TSP.txt' (see Appendix D).

2.5.3.2 Longitudinally Stiffened Panels

Limit states attributed to longitudinally stiffened panels are limit state 4 (yielding), limit state 2 (buckling), and limit state 3 (tripping).

2.5.3.2.1 Yielding (Limit State 4)

The yielding limit state of a longitudinally stiffened panel is defined as the yield stress, which is a material property. Like the other material properties in the Ultimate Strength program, yield stress of a longitudinally stiffened panel is calculated as a weighted average of yield stress values of its component plates and stiffener. The weight of each component is based on its cross-sectional area.

2.5.3.2.2 Buckling (Limit State 2)

The buckling limit state of a longitudinally stiffened panel is calculated by multiplying the ratio of the critical average stress to yield stress by the yield stress of the transversely stiffened panel.

The yield stress is calculated as a weighted average of yield stress values of its component plates and stiffener. The weight of each component is based on its area.

The ratio of critical collapse stress to yield stress is set by the CrossSection object when a SFTransverselyStiffenedPanel is created and by the HOODDocument object each time a SFTransverselyStiffenedPanel object is modified. The ratio is generated by an instance of class SSCP23Generator, which uses the value of the plate slenderness ratio and column slenderness ratio of the transversely stiffened panel in question to calculate it by bi-linearly interpolating data found in the file 'Average_Yield_Stress_LSP.txt' (see Appendix E).

2.5.3.2.3 Tripping (Limit State 3)

Calculations of the lateral buckling (tripping) limits state of stiffener are based on the dimensions of the longitudinally stiffened panel and its yield stress and make no use of data files.

The yield stress is calculated as a weighted average of yield stress values of its component plates and stiffener. The weight of each component is based on its area.

The formulae used to obtain the critical elastic buckling stress for stiffener tripping is provided in section 4 of *Incorporation of UK NS94 Ultimate Strength Analysis in MGDSA Part I: Engineering Considerations*.

2.5.3.3 Hard Corners and User-Defined Structures

There are no SSCP23 limit states attributed to hard corners and user-defined structures, but it is possible to obtain their yield stress values which are calculated as a weighted average of yield stress values of its component plates and stiffeners. The weight of each component is based on its area.

2.5.3.4 Calculating Component Stresses from Cross-section Bending Moments

Component stresses are based on the values of the applied horizontal and vertical bending moments. These values may be provided in MGDSA cross-section data files. DIGFEM data files do not contain such information.

Unless the applied bending moment data is available, no comparison may be made between SSCP23 limit states and the stress and therefore it is not possible to determine component failure. If applied bending moment data is available then component stresses are calculated. Only structural units that are active are considered in the calculations.

Since component stresses depend on curvature changes experienced by the entire cross-section and the cross-section is made up of many components, it follows that changes to the properties of any component will affect stresses experienced by all other

components. Accordingly, every time a component is modified, activated or deactivated, component stresses for the entire cross-section are re-calculated.

To determine if a CrossSection has been given values for the horizontal and vertical bending moments, use the hasBendingMoment() method passing it the constant value APPLIEDBM as the parameter value. Component stresses are set using the cross-section method setComponentStresses(). Other methods of interest are getNeutralSystemCenter(), getMoments(), and getCurvatureChanges(). For more information please see the internal documentation.

2.5.4 Running an Ultimate Strength Analysis

An ultimate strength analysis is run when the user selects the 'Run' menu item under the 'Analysis' heading in the main menu or when he or she clicks a corresponding toolbar button.

The generated message is handled by a method in the CUSDoc object called OnAnalysisRun(). This method uses the underlying HOODDocument object to generate input data. When finished, it builds command lines to be used for launching the appropriate analysis program and then launches the analysis programs and waits for its termination. Finally, it uses the HOODDocument object to read the ultimate strength results and displays them to the user.

Input data files for ultimate strength analysis programs generated by the UltimateStrength application and ultimate strength results files produced by those programs are all stored in the same directory as the Ultimate Strength project file ('.usp' file).

2.5.4.1 Generating Input Data for Ultimate Strength Analysis Programs

Input data for ultimate strength analysis programs can be generated only when the cross-section is fully defined and the program and its component databases are in good running state. Generating of input data for ultimate strength analysis programs ULTMAT, NS94A, NS94B, NS94C, and NS94D is done using writer objects, which are instances of classes ULTMATWriter, NS94AWriter, NS94BWriter, NS94CWriter, and NS94DWriter, respectively. These objects are created on the stack within protected HOODDocument methods writeULTMATInput(), writeNS94AInput(), writeNS94BInput(), writeNS94CInput(), and writeNS94Dinput, which are called through the public HOODDocument method writeAnalysisInput(). The selection of which method gets called depends on the currently selected analysis program.

All these methods have a common set of pre-conditions:

1) The CrossSection object must be fully defined (i.e. geometry, load-shortening curves, etc).
2) A valid directory where the files will be created must be provided.
3) A valid title for used within filenames must be provided.

2.5.4.1.1 Generating Input for ULTMAT

In the UltimateStrength application ULTMAT input is generated by calling the HOODDocument method writeULTMATInput(). This method performs the following tasks:

1) It concatenates the directory path and title parameters and the string "_ULTMAT.DAT" to form the full path to the ULTMAT input file.
2) It saves the resulting path within the HOODDocument object to be used later for finding the ULTMAT results file.

3) It builds a list of load-shortening curves, which are the non-interpolated curves that are stored in the VAST or NS94 LS-Curve database (as set by the user).
4) It creates an instance of ULTMATWriter, provides it with the path where the file should be generated, the load-shortening curve definitions and a pointer to the CrossSection object.
5) Finally, it calls the ULTMATWriter object's write() method to generate the ULTMAT input file.

The write() method of the writer object generates two files:

1) A file whose name terminates with "_ULTMAT.DAT" and which contains information about the geometry and its properties.
2) A "fem.dat" file, placed the same directory as other file, which contains the load-shortening curve definitions.

A detailed description of class ULTMATWriter can be found in its internal documentation.

2.5.4.1.2 Generating Input for NS94A

In the UltimateStrength application NS94A input is generated by calling the HOODDocument method writeNS94AInput(). This method performs the following tasks:

1) It concatenates the directory path and title parameters and the strings "_NS94A_HOG.DAT" and "_NS94A_SAG.DAT" to form two full paths, one to the NS94A input file that will produce a moment-curvature curve for the hogging loading condition, and the other to the NS94A input file that will produce a moment-curvature curve for the sagging loading condition.

2) It saves the resulting paths within the HOODDocument object to be used later for finding the NS94A results files.
3) It creates an instance of NS94AWriter, provides it with the path where the files should be generated, user-defined load ranges, if they were provided, and a pointer to the CrossSection object.
4) Finally, it calls the NS94AWriter object's write() method to generate the NS94A input file.

The write() method of the writer object generates two files:

1) A file whose name terminates with "_NS94A_HOG.DAT", which contains information for producing a moment-curvature curve for the hogging loading condition.
2) A file whose name terminates with "_NS94A_SAG.DAT", which contains information for producing a moment-curvature curve for the sagging loading condition.

A detailed description of class NS94AWriter can be found in its internal documentation.

2.5.4.1.3 Generating Input for NS94B

In the UltimateStrength application NS94B input is generated by calling the HOODDocument method writeNS94BInput().

If the selected analysis is set to generate a moment-curvature curve, then this method performs the following tasks:

1) It concatenates the directory path and title parameters and the strings "_NS94B_HOG.DAT" and "_NS94B_SAG.DAT" to form two full paths, one to the NS94B input file that will produce a moment-curvature curve for the hogging

loading condition, and the other to the NS94B input file that will produce a moment-curvature curve for the sagging loading condition.

2) It saves the resulting paths within the HOODDocument object to be used later for finding the NS94B results files.
3) It creates an instance of NS94BWriter, provides it with the path where the files should be generated, user-defined load ranges, if they were provided, and a pointer to the CrossSection object.
4) Finally, it calls the NS94BWriter object's write() method to generate the NS94B input files.

The write() method of the writer object generates two files:

1) A file whose name terminates with "_NS94B_HOG.DAT", which contains information for producing a moment-curvature curve for the hogging loading condition.
2) A file whose name terminates with "_NS94B_SAG.DAT", which contains information for producing a moment-curvature curve for the sagging loading condition.

If the selected analysis is set to generate an interaction curve then this method performs the following tasks:

1) It creates 28 full paths to NS94B input files by concatenating the directory path and title parameters and strings "_*N*_NS94B_INT.DAT", where *N* is replaced by a number ranging from 0 top 27.
2) It saves the resulting paths within the HOODDocument object to be used later for finding the NS94B results files.
3) It creates an instance of NS94BWriter, provides it with the path where the files should be generated, user-defined load ranges, if they were provided, and a pointer to the CrossSection object.

4) Finally, it calls the NS94BWriter object's write() method 28 times to generate 28 NS94B input files, each one for a specific curvature ratio.

The formats of these files are described in greater detail in document titled *Ultimate Strength of a Ship's Hull Under Biaxial Bending*. The curvature ratios that are used are described in Appendix H. A detailed description of class NS94BWriter can be found in its internal documentation.

2.5.4.1.4 Generating Input for NS94C

In the UltimateStrength application NS94C input is generated by calling the HOODDocument method writeNS94CInput(). This method performs the following tasks:

1) It concatenates the directory path and title parameters and the strings "_NS94C_HOG.DAT" and "_NS94C_SAG.DAT" to form two full paths, one to the NS94C input file that will produce a moment-curvature curve for the hogging loading condition, and the other to the NS94C input file that will produce a moment-curvature curve for the sagging loading condition.
2) It saves the resulting paths within the HOODDocument object to be used later for finding the NS94C results files.
3) It creates an instance of NS94CWriter, provides it with the path where the files should be generated, user-defined load ranges, if they were provided, and a pointer to the CrossSection object.
4) Finally, it calls the NS94CWriter object's write() method to generate the NS94C input files.

The write() method of the writer object generates two files:

1) A file whose name terminates with "_NS94C_HOG.DAT", which contains information for producing a moment-curvature curve for the hogging loading condition.
2) A file whose name terminates with "_NS94C_SAG.DAT", which contains information for producing a moment-curvature curve for the sagging loading condition.

The formats of these files are described in greater detail in document titled *NS94C/D Ultimate Strength Design Method for Ships*. A detailed description of class NS94CWriter can be found in its internal documentation.

2.5.4.1.5 Generating Input for NS94D

In the UltimateStrength application NS94D input is generated by calling the HOODDocument method writeNS94DInput(). This method performs the following tasks:

1) It concatenates the directory path and title parameters and the string "_NS94D_INT.DAT" to form the full path to the NS94D input file.
2) It saves the resulting path within the HOODDocument object to be used later for finding the NS94D results file.
3) It creates an instance of NS94DWriter, provides it with the path where the files should be generated and a pointer to the CrossSection object.
4) Finally, it calls the NS94DWriter object's write() method to generate the NS94D input files.

The write() method of the writer object generates a single file whose name terminates with "_NS94D_INT.DAT", which contains information for producing an interaction curve.

The formats of this file is described in greater detail in document titled *NS94C/D Ultimate Strength Design Method for Ships*. A detailed description of class NS94DWriter can be found in its internal documentation.

2.5.4.1.6 Executing Ultimate Strength Analysis Programs

In the Ultimate Strength application, analysis programs are executed from an instance of class CAnalysisProgressDlg, which is created on the stack in the CUSDoc OnAnalysisRun() method.

The CAnalysisProgressDlg object is ignorant of what it is actually launching and running. Its only responsibility is to start the processes and wait until they are finished. The responsibility for creating command lines lays within the OnAnalsisRun() method.

2.5.4.1.7 Creating Command Lines

Command lines for running ultimate strength analysis programs are created by taking the path of the selected analysis program and appending to it a command line argument, which is either the full path to an input file or a partial path to an input file that excludes the period and extension. Full paths are used with ULTMAT, while those that exclude the extension are used for NS94 programs.

Some programs and analyses require more than one run. In such cases multiple command lines are created.

2.5.4.1.8 Launching Ultimate Strength Analysis Programs

The first analysis run is launched when the first command line is executed in CAnalysisProgressDlg's OnInitDialog() override. Once the analysis process is running, the method sets up a timer that is fired every 500 milliseconds. Each time it triggers, it

invokes the method OnTimer(), which checks whether the analysis process has finished. If it has, then the OnTimer() methods checks whether there are more command lines to execute. If there is at least one, then the next command line is executed. This continues until all runs have finished. When done, the dialog closes and execution return to CUSDoc's OnAnalysisRun() method.

2.5.4.1.9 Reading Results from Ultimate Strength Analysis Programs

Ultimate strength analysis results are read in the OnAnalysisRun() method after all analysis runs have terminated. Reading is done using the HOODDocument object's public method readAnalysisResults(), which calls one of the protected methods readULTMATOutput(), readNS94AOutput(), readNS94BOutput(), readNS94COutput() or readNS94Doutput(), depending on the currently selected analysis program. These methods in turn make use of reader, which are instances of ULTMATReader, NS94AReader, NS94BReader, NS94CReader, and NS94DReader, respectively.

Paths to the results files generated by each ultimate strength analysis program are obtained using paths and file prefixes that were set when ultimate strength input data was generated with writeAnalysisInput(). All results files are stored in the same directory as the input files, which is the same directory as the Ultimate Strength project file ('.usp' file).

Results are stored in an internal HOODDocument attribute of class Results, which can be obtained by calling getAnalysisResults().

2.5.4.1.10 Reading Results From ULTMAT

In the Ultimate Strength application ULTMAT results are read by calling the HOODDocument method readULTMATOutput(). This method creates a path to the ULTMAT results file by appending the filename "moment.dat" to the directory path

where the ULTMAT input file was generated. Then it creates and instance of ULTMATReader, passes it the results file's path, and calls its read() method, which reads the results file's contents. The product or the reader are two curves, one representing the moment-curvature curve for the hogging loading condition and the other the moment-curvature curve for the sagging loading condition, which are assigned to an internal HOODDocument Results attribute that can be retrieved by calling getAnalysisResults().

2.5.4.1.11 Reading Results From NS94A

In the Ultimate Strength application NS94A results are read by calling the HOODDocument method readNS94AOutput(). This method creates two paths, one to the file containing results for the hogging loading condition and the other to the file containing results for the sagging loading condition. Each path is constructed by removing the '.DAT' extension from the corresponding input file and appending the '.PLT' extension. A NS94AReader is created and used to read curves from each output file. The curves are then assigned to an internal HOODDocument Results attribute that can be retrieved by calling getAnalysisResults().

2.5.4.1.12 Reading Results From NS94B

In the Ultimate Strength application NS94B results are read by calling the HOODDocument method readNS94BOutput(). This method reads moment-curvature curves as well as interaction curves.

If the current analysis selection is a moment-curvature curve, then the method creates two paths, one to the file containing results for the hogging loading condition and the other to the file containing results for the sagging loading condition. Each path is constructed by removing the '.DAT' extension from the corresponding input file and

appending the '.PLT' extension. A NS94BReader is created and used to read curves from each output file. The curves are then assigned to an internal HOODDocument Results attribute that can be retrieved by calling getAnalysisResults().

If the current analysis selection is an interaction curve, then the readNS94BOutput() method creates 28 paths, each to an output file with a specific curvature ratio. Each path constructed by removing the '.DAT' extension from the corresponding input file and appending the '.PLT' extension. A NS94BReader is created and used to read curves from each output file. The maximum horizontal and vertical bending moments from each file are used to generate the interaction curve, which is then assigned to an internal HOODDocument Results attribute that can be retrieved by calling getAnalysisResults().

2.5.4.1.13 Reading Results From NS94C

In the Ultimate Strength application NS94C results are read by calling the HOODDocument method readNS94COutput(). This method creates two paths, one to the file containing results for the hogging loading condition and the other to the file containing results for the sagging loading condition. Each path is constructed by removing the '.DAT' extension from the corresponding input file and appending the '.PLT' extension. A NS94CReader is created and used to read curves from each output file. The curves are then assigned to an internal HOODDocument Results attribute that can be retrieved by calling getAnalysisResults().

2.5.4.1.14 Reading Results From NS94D

In the Ultimate Strength application NS94D results are read by calling the HOODDocument method readNS94DOutput(). This method constructs a path to the NS94D results file by removing the '.DAT' extension from the corresponding input file and appending the '.INT' extension. A NS94DReader is created and used to read the

interaction curve from the results file, which is then assigned to an internal HOODDocument Results attribute that can be retrieved by calling getAnalysisResults().

2.5.4.1.15 Displaying Analysis Results

Analysis results are displayed automatically once an analysis has finished, but they can be also viewed at any time afterwards. Results are displayed by calling the HOODDocument::OnResultsPlots() method, which creates a property sheet with two property pages. The first page is an instance of class CResultsPlotsPage, which displays the results in terms of X-Y plots. The second page is an instance of class CResultsDataPage, which show the results in tabular form.

3 References

[1] Jiang Lei, Incorporation of UK NS94 Ultimate Strength Analysis in MGDSA Part I: Engineering Considerations (Martec TR-00-15), Martec Limited, Halifax, NS, Canada, February 2000.

[2] Norwood Merv and Wallace John, DSA-GP Graphics System User's Manual, Martec Limited, Halifax, NS, Canada, February 2000.

[3] Hu, S.Z. Thomas and Heath D.C., EBMS - A Computer Program for Equivalent Beam Modeling of Ship Structures, Defence Research Establishment Atlantic, Halifax NS, Canada.

[4] Grant R. M., Digitizer for Finite Element Modelling of Transverse Ship Sections and Planar Structures, Eyretechnics Limited, Dartmouth, NS, Canada, March 1989.

[5] Dow R. S. and Smith C. S., Ultimate Strength of a Ship's Hull Under Biaxial Bending, Admiralty Research Establishment, Dunfermline, UK, 1986.

[6] Dow R. S., NS94C/D Ultimate Strength Design Method for Ships, DRA, Dunfermline, UK, 1997.

[7] Martec Limited , Vibration and Strength Analysis Program (VAST): User's Manual Version 7.3, Martec Limited, Halifax, NS, Canada, April 2000.

[8] Chief Naval Architect, Sea Systems Controllerate Publication No. 23 (SSCP23), Design of Ship Structures Volume 1, Ministry of Defence, Foxhill, Bath, UK, December 1981.

[9] Chief Naval Architect, Sea Systems Controllerate Publication No. 23 (SSCP23), Design of Ship Structures Volume 2, Ministry of Defence, Foxhill, Bath, UK, December 1981.

[10] Dow R. S. and Smith C. S., FABSTRAN: A Computer Program for Frame and Beam Static and Transient Response Analysis (Nonlinear), Admiralty Research Establishment, Dunfermline, UK, 1986.

[11] WS Atkins Engineering Software, ASAS-NL User Manual Version 20 Update 4, WS Atkins Engineering Software, Surrey, England, 1998.

[12] Martec Limited, HOOD Reference Manual, Martec Limited, Halifax, NS, Canada February 1998.

[13] Jiang L, Wallace J, and Chernuka M.W. An Automatic Mesh Generator and Convergence Study for Calculating Load-Shortening Curves of Stiffened Plates for the Halifax Class Frigate (Martec TR-97-33), Martec Limited, Halifax, NS, Canada, October 1997.

APPENDIX A (ULTIMATE STRENGTH SYSTEM DIRECTORIES AND FILES)

This section lists the locations and descriptions of various directories and files used by the Ultimate Strength system. All paths are relative to the application's root directory <DIR>.

It is important that these locations and names remain consistent. The current implementation relies on this setup for proper program execution.

Directory	Description
<DIR>\	Root directory for the Ultimate Strength application containing the executable and other files and sub-directories.
<DIR>\Projects\	Default directory for storing projects.
<DIR>\SSCP23\	Contains data files used in calculating SSCP23 limit states
<DIR>\US HOOD DBs\	Contains HOOD database files
<DIR>\US LSC DBs\	Contains directories with LS-Curve databases
<DIR>\US LSC DBs\NS94D DB\	Contains the NS94D LS-Curve database
<DIR>\US LSC DBs\SPECIAL DB\	Contains the Special LS-Curve database
<DIR>\US LSC DBs\VAST DB\	Contains directories comprising the VAST LS-Curve database
<DIR>\US LSC DBs\VAST DB\c11\	Contains VAST LS-Curve files for the center-to-center model and stiffener area-to-stiffened panel area ratio of 0.2 and coefficient of plate imperfection of 0.1.
<DIR>\US LSC DBs\VAST DB\c12\	Contains VAST LS-Curve files for the center-to-center model and stiffener area-to-stiffened panel area ratio of 0 2 and coefficient of plate imperfection of 0.2.
<DIR>\US LSC DBs\VAST DB\c13\	Contains VAST LS-Curve files for the center-to-center model and stiffener area-to-stiffened panel area ratio of 0.2 and coefficient of plate imperfection of 0.3.
<DIR>\US LSC DBs\VAST DB\c21\	Contains VAST LS-Curve files for the center-to-center model and stiffener area-to-stiffened panel area ratio of 0.275 and coefficient of plate imperfection of 0.1.
<DIR>\US LSC DBs\VAST DB\c22\	Contains VAST LS-Curve files for the center-to-center model and stiffener area-to-stiffened panel area ratio of 0.275 and coefficient of plate imperfection of 0.2.
<DIR>\US LSC DBs\VAST DB\c23\	Contains VAST LS-Curve files for the center-to-center model and stiffener area-to-stiffened panel area ratio of 0.275 and coefficient of plate imperfection of 0.3.
<DIR>\US LSC DBs\VAST DB\c31\	Contains VAST LS-Curve files for the center-to-center model and stiffener area-to-stiffened panel area ratio of 0.35 and coefficient of plate imperfection of 0.1.
<DIR>\US LSC DBs\VAST DB\c32\	Contains VAST LS-Curve files for the center-to-center model and stiffener area-to-stiffened panel area ratio of 0.35 and coefficient of plate imperfection of 0.2.
<DIR>\US LSC DBs\VAST DB\c33\	Contains VAST LS-Curve files for the center-to-center model and stiffener area-to-stiffened panel area ratio of 0.35 and coefficient of plate imperfection of 0.3.
<DIR>\US LSC DBs\VAST DB\e11\	Contains VAST LS-Curve files for the end-to-end model and stiffener area-to-stiffened panel area ratio of 0.2 and coefficient

	of plate imperfection of 0.1.
<DIR>\US LSC DBs\VAST DB\e12\	Contains VAST LS-Curve files for the end-to-end model and stiffener area-to-stiffened panel area ratio of 0.2 and coefficient of plate imperfection of 0.2.
<DIR>\US LSC DBs\VAST DB\e13\	Contains VAST LS-Curve files for the end-to-end model and stiffener area-to-stiffened panel area ratio of 0.2 and coefficient of plate imperfection of 0.3.
<DIR>\US LSC DBs\VAST DB\e21\	Contains VAST LS-Curve files for the end-to-end model and stiffener area-to-stiffened panel area ratio of 0.275 and coefficient of plate imperfection of 0.1.
<DIR>\US LSC DBs\VAST DB\e22\	Contains VAST LS-Curve files for the end-to-end model and stiffener area-to-stiffened panel area ratio of 0.275 and coefficient of plate imperfection of 0.2.
<DIR>\US LSC DBs\VAST DB\e23\	Contains VAST LS-Curve files for the end-to-end model and stiffener area-to-stiffened panel area ratio of 0.275 and coefficient of plate imperfection of 0.3.
<DIR>\US LSC DBs\VAST DB\e31\	Contains VAST LS-Curve files for the end-to-end model and stiffener area-to-stiffened panel area ratio of 0.35 and coefficient of plate imperfection of 0.1.
<DIR>\US LSC DBs\VAST DB\e32\	Contains VAST LS-Curve files for the end-to-end model and stiffener area-to-stiffened panel area ratio of 0.35 and coefficient of plate imperfection of 0 2.
<DIR>\US LSC DBs\VAST DB\e33\	Contains VAST LS-Curve files for the end-to-end model and stiffener area-to-stiffened panel area ratio of 0.35 and coefficient of plate imperfection of 0.3.
<DIR>\US Programs\	Contains programs for doing ultimate strength and finite element analyses
<DIR>\ US Temporary Files\	Contains intermediate files generated during a FEA analysis with VAST and ASAS-NL
C:\USTemp	Contains FABSTRAN input and output files and intermediate results

<DIR>

File	Description
UltimateStrength.exe	Ultimate Strength program executable
ULTIMATESTRENGTH.GID	
ULTIMATESTRENGTH.HLP	Ultimate Strength help file
ultimatestrength.cnt	Ultimate Strength help contents file
ultimatesetrength.avi	Animation file used in dialogs etc.
401comupd.exe	Internet Explorer 4.01 ActiveX controls update
Axdist.exe	Chart FX 3.0 ActiveX control distribution/setup

<DIR>\SSCP23\

File	Description
Average_Yield_Stress_LSP.txt	File with column strength curves average imperfections used in SSCP23 limit state calculations for longitudinally stiffened panels
Average_Yield_Stress_TSP.txt	File with average critical yield stress to yield stress ratios used in SCCP23 limit state calculations for transversely stiffened panels

<DIR>\US HOOD DBs\

File	Description
HOOD_BEAM_DB.HDB	Main beam definition file
HOOD_BEAM_DB_IMPERIAL.HDB	Beam definition file for beams defined in imperial units (included

	into HOOD_BEAM_DB.HDB, but not used by the application)
HOOD_BEAM_DB_METRIC.HDB	Beam definition file for beams defined in metric units (included into HOOD_BEAM_DB.HDB)
HOOD_DAMAGE_DB.HDB	Damage definition file
HOOD_MATERIAL_DB.HDB	Material definition file

<DIR>\US LSC DBs\NS94D DB\

File	Description
NS94D_LCurves.dat	NS94D LS-Curves for longitudinally stiffened panels
NS94D_TCurves.dat	NS94D LS-Curves for transversely stiffened panels

<DIR>\US LSC DBs\SPECIAL DB\

File	Description
Lscurve.mfl	Master file list for the Special LS-Curve database containing filename/curve title pairs. These pairs are used for obtaining a filename given a load-shortening curve's title. (Note that this file may be empty, but it must exist.)
LSC*ddddddddd*.LSC	Files containing load-shortening curve definitions. '*ddddddddd*' represents a unique eight digit number ranging from 00000000 to 99999999. The directory may contain up to 100,000,000 unique files/curve definitions. These files must exist only if they are listed in 'Lscurve.mfl'.

<DIR>\US LSC DBs\VAST DB*mcd*\

File	Description
mabcd.dat	VAST database file for a given model (*m*), plate slenderness ratio (*a*) , column slenderness ratio (*b*), stiffener area to stiffened panel area ratio (*c*), and maximum coefficient of plate imperfection (*d*) 'm' is either 'c' for center-to-center model or 'e' for end-to-end model. 'a' is a digit ranging from 1 to 4, representing plate slenderness ratios of 1.0, 2.0, 3.0, and 4.0 respectively. 'b' is a digit ranging from 1 to 4, representing column slenderness ratios of 0.3, 0.6, 0.9, and 1.2 respectively. 'c' is a digit ranging from 1 to 3, representing stiffener are to stiffened panel area ratio of 0.2, 0 275, and 0.35 respectively. 'd' is a digit ranging from 1 to 3, representing the maximum coefficient of plate imperfection of 0 1, 0.2, and 0.3 respectively.

<DIR>\US Programs\

File	Description
Ns94a exe	NS94A executable (Built on 08/07/99 @ 10:03 AM)
Ns94b.exe	NS94B executable (Built on 20/10/99 @ 10:32 AM)
Ns94c.exe	NS94C executable (Built on 09/11/99 @ 3:37 PM)
Ns94d.exe	NS94D executable (Built on 27/09/99 @ 2:42 PM)
ULTMAT.exe	ULTMAT executable (Built on 08/11/99 @ 12:15 PM)
LSPanel.exe	LSPanel executable (Built on 30/09/99 @ 9:35 AM)
vast.exe	VAST executable (Built on 19/10/99 @ 1:32 PM)
FABSTRAN.exe	FABSTRAN executable (Built on 16/08/99 @ 11:15 AM)
prenl20 exe	PRENL 2.0 executable (goes with asasnl20.exe)
asasnl20.exe	ASAS-NL 2.0 executable (goes with prenl20.exe)

*Note that prenl20 exe and asasnl20.exe must be placed in the directory by the user as the Ultimate Distribution does not contain these files

<DIR>\ US Temporary Files\

File	Description
All files	Files used by and produced by VAST and ASAS-NL. For file descriptions consult manuals for VAST and ASAS-NL

C:\USTemp

File	Description
Fabstran.bat	Batch file produced to set up and run FABSTRAN FE analysis.
Other files	Files used by and produced by FABSTRAN. For file descriptions consult the FABSTRAN manual.

APPENDIX B (PROJECT FILES)

Introduction

Each Ultimate Strength project consists of four files made up of a prefix and one of the extensions '.usp', '.hod', '.hdb', '.rlt'.

The Prefix

The prefix is specified by the user when the user creates a new project. In addition to being used within names given to project files, the prefix is used to name the directory where the files reside. This organization makes it easier to distinguish between projects. Whenever possible, it is also used as a prefix for input, output, and intermediate files necessary for running various programs like ULTMAT and VAST.

The USP File

A '.usp' is a binary file and serves as the main project file. It contains project version numbers and project options. See internal documentation for file format details.

The HOD File

A '.hod' file contains the definition of a cross-section (primarily its geometry) which is represented in terms of an instance of class CrossSection. Class CrossSection is described briefly in Appendices F and G. A detailed description can be found in the internal documentation.

The HDB File

A '.hdb' file contains the definitions of load-shortening curves particular to the cross-section defined in the '.hod' file. The contents of this file comprise a definition of the scene graph, which serves as the load-shortening curve database. The nodes in this scene graph are instances of classes LSCurve, LSCurveLSP, and LSCurveTSP, which represent curves for hard-corners, longitudinally stiffened panels, and transversely stiffened panels respectively. See internal documentation for file format details.

The RLT File

A '.rlt' file contains results of the previously performed analysis. This file exists even if no analysis has been performed. Results are stored in terms of an instance of class Results. See internal documentation for file format details.

APPENDIX C (MGDSA CROSS-SECTION FILE FORMAT)

This appendix contains a brief description of the MGDSA '.usd' file format and a sample file containing a definition of a plate and a stiffener.

Literal values have been italicized. Note that each field has an assigned width but here filed widths have not been specified. Instead, data items are separated by white-space.

File Contents (Lines)	Records
COORDINATE TRANSFORMATION DATA	Record 1
IFWDUS IPSUS IVFRTUS	Record 2
DIRCOS1 DIRCOS2 DIRCOS3	Record 3
DIRCOS4 DIRCOS5 DIRCOS6	
DIRCOS7 DIRCOS8 DIRCOS9	
IUNITL IUNITF	Record 4
SECTION DEFINITION DATA	Record 5
XSLUS YSLUS ZSLUS	Record 6
FRAMENO EYE	Record 7
NUMCOM	Record 8
COMID X1 Y1 Z1 X2	Record 9
Y2 Z2 L E V	
S TISECT W D TF	
TW	
D1 D2 D3 D4 D5	Record 10
D6 D7 D8	
BENDING MOMENT	Record 11
HBM VBM	Record 12

Record	Description
1	ASCII string representing a key-word (ignored)
2	Integer values 1, -1, 2, -1, 3, -3 that relate the finite element model coordinate system to forward (IFWDUS) direction, portside(IPSUS) direction and vertical (IVERTUS) direction. Values 1 and –1 indicate that the direction is towards the positive X axis or the negative X axis respectively. Values 2 and –2 indicate that the direction is towards the positive Y axis or the negative Y axis respectively. Values 3 and –3 indicate that the direction is towards the positive Z axis or the negative Z axis respectively.
3	Double values representing Direction cosines (ignored)
4	Integer values specifying the units of length (IUNITL) and the units of force (IUNITF). IUTIL values can be 0, 1, 2, 3, 4, indicating that the units of length are not specified, in inches, in feet, in millimeters, or in meters. IUNITF values can be 0, 1, 2, 3, indicating that the units of force are not specified, in pounds, in kilo-Pascals, or in Newtons.
5	ASCII string representing a key-word (ignored)
6	Double values representing an arbitrary point on the cross-section. XSLUS, YSLUS, and ZSLUS are its x, y, and z coordinates. The coordinate is specified in the finite element model coordinate system.
7	Integer values. FRAMNO represents the frame number. EYE can be 1 or –1 indicating that the structure is in the front of the frame or in the back of the frame.
8	Integer value, NUMCOM, indicates the number of components.
9	This record consists of 15 doubles, which define a component. It is repeated NUMCOM times. The type of component is identified by the value of COMID. Value of 1 indicates

	a plate. Value of 2 indicates a beam. The interpretation of other fields depends on the component type. For plates, X1, Y1, and Z1 define the coordinate defining the start of the plate and X2, Y2, and Z2 define the coordinate of the end of the plate. For stiffeners X1, Y1, and Z1 define the coordinate of the base of the web and X2, Y2, and Z2 serve as a reference coordinate for determining the beam direction. The next four fields, L, E, V, S are common to both plates and beams and represent the length, modulus of elasticity, Poisson's ratio, and yield stress. For plates, the following field, TISECT, represents the plate thickness and the remaining four fields are ignored. For stiffeners, TISECT, represents the beam type. Its value may be 1, 7, or 10, indicating a flat-bar, angle, or T beam. The remaining parameters, W, D, TF, and TW, represent the flange width, web depth, flange thickness and web thickness, respectively.
10	Eight double values. (ignored)
11	ASCII string representing a key-word. If this string is not found, then it can be assumed that no applied bending moment data is provided.
12	Two double fields, HBM and VBM, representing the applied horizontal and vertical bending moments.

Sample File

This file is fictitious and is provided to show the general layout of a MGDSA USD file.

```
COORDINATE TRANSFORMATION DATA
          -1            3            2
  -1.000000      0.0000000E+00  0.0000000E+00
  0.0000000E+00  0.0000000E+00   1.000000
  0.0000000E+00   1.000000      0.0000000E+00
           3            3
SECTION DEFINITION DATA
   18000.00       12160.00      -7132.000
           2            1
           2
   1.000000       17865.68       12224.30       6080.942       17865.68
   12278.58       4980.942       1000.000       210000.0       0.3000000
   350.0000       8.000000      0.0000000E+00  0.0000000E+00  0.0000000E+00
  0.0000000E+00
   2.000000       17865.68       12251.44       5530.942       17865.68
   12124.60       5524.684       1000.000       210000.0       0.3000000
   350.0000       10.00000       68.00000       127.0000       5.200000
   4.000000
699846.5      -0.8064191       6841.102      1.1522643E+13  7.9611319E+12
  0.0000000E+00  0.0000000E+00  0.0000000E+00
BENDING MOMENT
 3.000E+11  3.000E+11
```

APPENDIX D (FILE AVERAGE_YIELD_STRESS_TSP.TXT)

```
September 30, 1999: Average Critical Yield Stress to Yield Stress Ratios (
Beta, sigma_a/ sigma_y )
{
1.0     0.9911,
1.5     0.8820,
2.0     0.7370,
2.5     0.6036,
3.0     0.5179,
3.5     0.4625,
4.0     0.4130
}
```

APPENDIX E (AVERAGE_YIELD_STRESS_LSP.TXT)

#Column Strength Curves Average Imperfections
#x-axis =lambda, y-axis = sigA/sigY
#File Format = x, y

Beta= 1 0 23
0.14500485 0.878940731
0 192919496 0.878940731
0.243355965 0.880201765
0.292531523 0.880201765
0.344228904 0.880201765
0.394665373 0.877679697
0 45014549 0 873896595
0.496799224 0.868852459
0.549757517 0.860025221
0.60271581 0 852459016
0 653152279 0.842370744
0.709893307 0.829760404
0.757807953 0.815889029
0.798157129 0.802017654
0 84985451 0.780580076
0 897769156 0.757881463
0.945683802 0 733921816
0.989815713 0.711223203
1 038991271 0.68221942
1 085645005 0.655737705
1.12599418 0.631778058
1.163821532 0.607818411
1.209214355 0.578814628
Beta= 1.5 21
0 148787585 0 803278689
0.200484966 0.803278689
0 249660524 0.805800757
0.298836081 0.805800757
0.350533463 0.804539723
0.411057226 0.802017654
0 469059166 0.796973518
0.529582929 0 790668348
0.585063046 0 783102144
0 635499515 0.773013871
0.692240543 0.762925599
0.74772066 0 749054224
0.810766246 0.730138714
0.870029098 0.709962169
0.917943744 0.689785624
0.969641125 0 665825977
1.022599418 0.640605296
1.074296799 0.614123581
1.124733269 0.587641866
1.161299709 0.567465322
1.207953443 0.543505675
Beta= 2.0 24
0.146265761 0.718789407
0.185354025 0.720050441

0.225703201 0.720050441
0.273617847 0.720050441
0 32657614 0.720050441
0.374490786 0 718789407
0 418622696 0.717528373
0.475363725 0.715006305
0.529582929 0.711223203
0.574975752 0.706179067
0.621629486 0.698612863
0.673326867 0.691046658
0.716197866 0.68221942
0.767895247 0.670870113
0.824636275 0.656998739
0.873811833 0.643127364
0.912900097 0 627994956
0.960814743 0.612862547
1.007468477 0 595208071
1.049078565 0.57629256
1.09699321 0.556116015
1.138603298 0 538461538
1 171387003 0 524590164
1.206692532 0.508196721
Beta= 2 5 20
0 151309408 0.62925599
0.209311348 0 62925599
0.259747818 0 62925599
0 312706111 0.62925599
0 369447139 0.627994956
0 423666343 0 627994956
0.480407371 0.62295082
0 549757517 0.617906683
0.614064016 0.610340479
0 669544132 0 601513241
0.723763337 0.592686003
0 780504365 0.58259773
0.829679922 0.571248424
0 883899127 0.558638083
0.934335597 0.544766709
0.988554801 0.528373266
1 050339476 0.510718789
1.112124151 0.490542245
1.161299709 0.474148802
1 21299709 0 455233291
Beta= 3.0 20
0 155092144 0.552332913
0.213094083 0.552332913
0.262269641 0 552332913
0.325315228 0.552332913
0.387099903 0.549810845
0.448884578 0.548549811
0.506886518 0.544766709
0.557322987 0.540983607
0.617846751 0.534678436
0 682153249 0.527112232
0.746459748 0.518284994
0.803200776 0.511979823

0.858680892 0.501891551
0.921726479 0.488020177
0.984772066 0.472887768
1.042774006 0.457755359
1.094471387 0.445145019
1.14742968 0.431273644
1 186517944 0.419924338
1.226867119 0.408575032
Beta= 3.5 18
0.156353055 0 500630517
0.20300679 0.500630517
0.253443259 0.499369483
0.302618817 0.498108449
0.360620757 0.496847415
0.414839961 0.495586381
0.482929195 0.491803279
0.540931135 0.486759142
0 609020369 0.481715006
0.672065955 0.475409836
0.742677013 0.466582598
0 827158099 0.460277427
0.890203686 0 450189155
0.959553831 0.438839849
1.021338506 0 426229508
1.08942774 0 413619168
1.151212415 0.399747793
1.20543162 0.388398487
Beta= 4.0 18
0 155092144 0.459016393
0 213094083 0.459016393
0.262269641 0 459016393
0 321532493 0 460277427
0.387099903 0 457755359
0 45771096 0 457755359
0.518234724 0 455233291
0.586323957 0.452711223
0.641804074 0.447667087
0.701066925 0.443883985
0 761590689 0.440100883
0.818331717 0.431273644
0.892725509 0.422446406
0.969641125 0.4110971
1.037730359 0 399747793
1.102036857 0 387137453
1.158777886 0 37704918
1.206692532 0.366960908

APPENDIX F (SERIALIZATION/FORMATS OF OBJECTS)

This section contains the format of the objects that define a cross-section specified using the standard HOOD file format notation. Objects that were used from the HOOD toolkit are not documented.

SFKeyPoint

A SFKeyPoint object is serialized in terms of its ID, followed by an X-Y position coordinate.

```
<SFInt32> <SFVec2f>
```

SFPlate

A SFPlate object is serialized in terms of its ID, followed by the IDs of the two plates that defined its width, followed by the thickness, the length, the ID of the plate material, and the ID of the associated damage definition.

```
<SFInt32> <SFInt32> <SFInt32> <SFFloat> <SFFloat> <SFString> <SFString>
```

SFStiffener

A SFStiffener object is serialized in terms of its ID, followed by the ID of the key-point to which the stiffener is attached, followed by the X-Y coordinate of a point used to set its orientation, followed by the length, the ID of the stiffener definition from the stiffener kind database, the ID of the plate material, and the ID of the associated damage definition.

```
<SFInt32> <SFInt32> <SFVec2f> <SFFloat> <SFString> <SFString> <SFString>
```

SFTransverselyStiffenedPanel

A SFTransverselyStiffenedPanel object is serialized in terms of its ID, followed by the IDs of the two plates that comprise it, followed by the ID of the associated load-shortening curve, followed by a flag indicating whether it is active or inactive.

```
<SFInt32> <SFInt32> <SFInt32> <SFString> <SFBool>
```

SFLongitudinallyStiffenedPanel

A SFLongitudinallyStiffenedPanel object is serialized in terms of its ID, followed by the ID of the stiffener, followed by the IDs of the two plates, followed by the ID of the associated load-shortening curve, followed by a flag indicating whether it is active or inactive.

```
<SFInt32> <SFInt32> <SFInt32> <SFInt32> <SFString> <SFBool>
```

SFHardCorner

A SFHardCorner object is serialized in terms of its ID, followed by a list of IDs of plates that comprise it, a list of stiffeners that comprise it, followed by the ID of the associated load-shortening curve, followed by a flag indicating whether it is active or inactive.

```
<SFInt32> <MFInt32> <MFInt32> <SFString> <SFBool>
```

SFUserDefinedStructure

A SFUserDefinedStructure object is serialized in terms of its ID, followed by a list of IDs of key-points that comprise it, followed by a list of IDs of plates that comprise it, a list of stiffeners that comprise it, followed by the ID of the associated load-shortening curve, followed by a flag indicating whether it is active or inactive.

```
<SFInt32> <MFInt32> <MFInt32> <MFInt32> <SFString> <SFBool>
```

CrossSection

A CrossSection is serialized in terms of its properties: description (not used), units of length, units of force, viewing end, location, station ID, flag indicating whether applied bending moments are available, applied horizontal and vertical bending moments, flag indicating whether maximum bending moments are available (not used), maximum horizontal and vertical bending moments, a list of key-points, a list of plates, a list of stiffeners, a list of longitudinally stiffened panels, a list of transversely stiffened panels, a list of hard corners, and a list of user-defined structures.

```
DEF Cross-section CrossSection {
      Description          <SFString>
      UnitsOfLength        <SFString>
      UnitsOfForce         <SFString>
      ViewingEnd           <SFString>
      Location             <SFFloat>
      StationID            <SFInt32>
      HasAppliedBM         <SFBool>
      AppliedHBM           <SFFloat>
      AppliedVBM           <SFFloat>
      HasMaxBM             <SFBool>
      MaxHBM               <SFFloat>
      MaxVBM               <SFFloat>
      KeyPoints            <MFKeyPoint>
      Plates               <MFPlate>
      Stiffeners           <MFStiffener>
```

```
        LSPs            <MFLongitudinallyStiffenedPanel>
        TSPs            <MFTransverselyStiffenedPanel>
        HCs             <MFHardCorner>
        UDSs            <MFUserDefinedStructure>
}
```

APPENDIX G (IMPORTANT CLASSES)

Introduction

This appendix provides a brief description of the most important classes that comprise the Ultimate Strength project.

CUSApp

The CUSApp class is derived from the MFC class CWinApp. Instances of this class contain application specific attributes such as paths to ultimate strength analysis programs (ULTMAT, NS94A, NS94B, NS94C, and NS94D), paths to finite element analysis programs (VAST, FABSTRAN and ASAS-NL), path of LSPanel, temporary directory path, as well as material, damage, and stiffener kind databases, the SSCP23 ratio generator, and load-shortening curve databases (VAST, NS94D, and the Special LS-Curve DB).

CUSDoc

The CUSDoc class is derived from the MFC class CDocument. Instances of this class contain document specific attributes such as the HOODDocument object (the HOOD document) and methods for reading and writing cross-section information and for handling changes the model.

CUSView

The CUSView class is derived from the MFC class CView. Instances of this class contain the HOOD view object and are responsible for the presentation and rendering of the model and for displaying most of Ultimate Strength dialogs.

HOODDocument

The HOODDocument class is derived from HOOD class Application. Instances of this class contain document specific attributes, like the CrossSection object, the load-shortening curve generator, the load-shortening curve database with curves particular to the cross-section, analysis results, project options, as well as references to the various databases (which in the UltimateStrength application are contained in the single CUSApp object). This class is responsible for importing cross-section definitions from MGDSA and DIGFEM, generating and assigning load-shortening curves, for generating input for NS94 and ULTMAT programs, and for handling events affecting the cross-section model and project properties.

CrossSection

The CrossSection class is derived from HOOD class SFGNode. Instances of this class contain attributes that are specific to a model of a cross-section. Among these attributes are lists of geometric elements like key-points, plates, stiffeners, longitudinally stiffened panels, transversely stiffened panels, hard corners, user defined structures, references to the material, damage, and stiffener kind databases, and attributes that deal with model presentation. In addition, the CrossSection class contains methods that are responsible for analyzing geometry as well as calculating various global properties that are based on component objects.

SFKeyPoint

The SFKeyPoint class is derived from the HOOD class SFVec2f and the class Selectable. Instance of this class represent key-points, which are defined as plate-plate and plate-stiffener junctions. SFKeyPoint objects contain attributes that keep track of the associated plate and stiffener objects.

SFPlate

The SFPlate class is derived from the class SFBasicStructure. Instances of this class represent plates. They contain attributes that define plate dimensions, material, and damage, as well as attributes that keep track of associated SFKeyPoint, SFHardCorner, SFLongitudianllyStiffenedPanel, and SFTransverselyStiffenedPanel objects. In addition, class SFPlate contains methods for calculating moments of inertia.

SFStiffener

The SFStiffener class is derived from the class SFBasicStructure. Instances of this class represent stiffeners (T-Beams). They contain attributes that define stiffener dimensions, material, and damage, as well as attributes that keep track of associated SFKeyPoint, SFHardCorner, and SFLongitudianllyStiffenedPanel objects. In addition, class SFStiffener contains methods for calculating moments of inertia.

SFLongitudinallyStiffenedPanel

The SFLongitudinallyStiffenedPanel class is derived from class SFStiffenedPanel. Instances of this class represent longitudinally stiffened panels. They contain attributes that reference the two plates and stiffener that form the panel, as well as various methods for calculating properties like moments of inertia, plate and column slenderness ratios, and SSCP23 limit states.

SFTransverslyStiffenedPanel

The SFTransverselyStiffenedPanel class is derived from class SFStiffenedPanel. Instances of this class represent transversely stiffened panels. They contain attributes that reference the two plates that form the panel, as well as various methods for

calculating properties like moments of inertia, column slenderness and panels aspect ratios, and SSCP23 limit states.

SFHardCorner

The SFHardCorner class is derived from class SFStructuralUnit. Instances of this class represent hard corners. They contain attributes that reference plates and stiffeners that form the panel, as well as various methods for calculating properties like moments of inertia.

SFUserDefinedStructure

The SFUserDefinedStructure class is derived from class SFStructuralUnit. Instances of this class represent structural units defined by the user. They contain attributes that reference plates and stiffeners that form the panel, as well as various methods for calculating properties like moments of inertia.

MGDSAReader

The class MGDSAReader is responsible for reading cross-section definition files that are in MGDSA format.

NS94AWriter

The class NS94AWriter is responsible for translating a CrossSection object into file format used for NS94A input files.

NS94BWriter

The class NS94BWriter is responsible for translating a CrossSection object into file format used for NS94B input files.

NS94CWriter

The class NS94CWriter is responsible for translating a CrossSection object into file format used for NS94C input files.

NS94DWriter

The class NS94DWriter is responsible for translating a CrossSection object into file format used for NS94D input files.

ULTMATWriter

The class ULTMATWriter is responsible for translating a CrossSection object into file format used for ULTMAT input fules.

NS94AReader

The class NS94AReader is responsible for reading ultimate strength results produced by NS94A.

NS94BReader

The class NS94BReader is responsible for reading ultimate strength results produced by NS94B.

NS94CReader

The class NS94CReader is responsible for reading ultimate strength results produced by NS94C.

NS94DReader

The class NS94DReader is responsible for reading ultimate strength results produced by NS94D.

ULTMATReader

The class ULTMATReader is responsible for reading ultimate strength results produced by ULTMAT.

LSPanelWriter

The class LSPanelWriter is responsible for generating the LSPanel input file.

VASTReader

The class VASTLSCurveReader is responsible for reading a load-shortening curve definition produced via VAST FE analysis.

FABSTRANLSCurveReader

The class FABSTRANLSCurveReader is responsible for reading a load-shortening curve definition produced via FABSTRAN FE analysis.

ASASNLLSCurveReader

The class ASASNLLSCurveReader is responsible for reading a load-shortening curve definition produced via ASAS-NL FE analysis.

VASTDBLSCReader

The class VASTDBLSCReader is responsible for reading load-shortening curve definitions found in the VAST LS-Curve database.

NS94DLSCReader

The class NS94DLSCReader is responsible for reading load-shortening curve definitions found in the NS94D LS-Curve database.

SpecialLSCDBInterface

The class SpecialLSCDBInterface is responsible for reading load-shortening curve definitions found in the Special LS-Curve database, which contains curves generated by the user.

SSCP23RatioGenerator

The class SSCP23RatioGenerator is responsible for generating critical average stress to yield stress ratios for transversely and longitudinally stiffened panels.

NS94DCurveGenerator

The class NS94DCurveGenerator is responsible for generating load-shortening curves for transversely and longitudinally stiffened panels based on curves stored in the NS94D LS-Curve database.

LSCurveGenerator

The class LSCurveGenerator is responsible for generating load-shortening curves for longitudinally stiffened panels based on curves stored in the VAST LS-Curve database.

GeneralLSCurveGenerator

The class GeneralLSCurveGenerator is responsible for generating load-shortening curves for stiffened panels using the NS94D and VAST LS-Curve databases.

Interpolator

The class Interpolator is responsible for performing bi-linear or linear-exponential interpolation on four curves.

APPENDIX H (NS94B INTERACTION CURVE RATIOS)

The following is a list of curvature ratios (My:Mz) used for generating input for NS94B when calculating a bi-axial interaction curve.

	My (horizontal)	Mz (vertical)
1	1.0E-02,	1.0
2	3.0,	20.0
3	1.0,	3.0
4	2.0,	3.0
5	1.0,	1.0
6	3.0,	2.0
7	3.0,	1.0
8	1.0E+02,	1.0
9	3 0,	-1.0
10	3 0,	-2.0
11	1.0,	-1.0
12	2 0,	-3 0
13	1.0,	-3.0
14	3 0,	-20.0
15	1 0E-02,	-1.0
16	-3 0,	-20.0
17	-1.0,	-3.0
18	-2 0.	-3.0
19	-1 0,	-1.0
20	-3.0,	-2 0
21	-3.0,	-1.0
22	-1 0E+02,	-1.0
23	-3 0,	1 0
24	-3.0,	2.0
25	-1.0,	1.0
26	-2.0,	3.0
27	-1.0,	3.0
28	-3.0,	20.0

UNCLASSIFIED
SECURITY CLASSIFICATION OF FORM
(highest classification of Title, Abstract, Keywords)

DOCUMENT CONTROL DATA

(Security classification of title, body of abstract and indexing annotation must be entered when the overall document is classified)

1 ORIGINATOR (the name and address of the organization preparing the document Organizations for whom the document was prepared, e.g. Establishment sponsoring a contractor's report, or tasking agency, are entered in section 8.)

Martec Ltd., Suite 400, 1888 Brunswick St., Halifax, NS, B3J 3J8

2. SECURITY CLASSIFICATION (overall security classification of the document including special warning terms if applicable).

UNCLASSIFIED

3 TITLE (the complete document title as indicated on the title page. Its classification should be indicated by the appropriate abbreviation (S,C,R or U) in parentheses after the title).

Incorporation of UK NS94 Ultimate Strength Analysis in MGDSA, Part II: Software Considerations

4 AUTHORS (Last name, first name, middle initial. If military, show rank, e.g. Doe, Maj John E.)

Lichodzijewski, M., and Jiang, L.

5 DATE OF PUBLICATION (month and year of publication of document)	6a NO OF PAGES (total containing information Include Annexes, Appendices, etc).	6b NO OF REFS (total cited in document)
March 2000	83 (approx.)	6

7 DESCRIPTIVE NOTES (the category of the document, e.g. technical report, technical note or memorandum If appropriate, enter the type of report, e.g. interim, progress, summary, annual or final Give the inclusive dates when a specific reporting period is covered)

CONTRACTOR REPORT

8 SPONSORING ACTIVITY (the name of the department project office or laboratory sponsoring the research and development Include address)

Defence Research Establishment Atlantic
PO Box 1012
Dartmouth, NS, Canada B2Y 3Z7

9a PROJECT OR GRANT NO (if appropriate, the applicable research and development project or grant number under which the document was written Please specify whether project or grant)	9b. CONTRACT NO (if appropriate, the applicable number under which the document was written)
1 g.c.15	W8477-8-0004/001/SV

10a ORIGINATOR'S DOCUMENT NUMBER (the official document number by which the document is identified by the originating activity This number must be unique to this document)	10b OTHER DOCUMENT NOs (Any other numbers which may be assigned this document either by the originator or by the sponsor.)
TR-00-13	DREA CR 2000-113

11 DOCUMENT AVAILABILITY (any limitations on further dissemination of the document, other than those imposed by security classification)

(x) Unlimited distribution
( ) Defence departments and defence contractors; further distribution only as approved
( ) Defence departments and Canadian defence contractors; further distribution only as approved
( ) Government departments and agencies; further distribution only as approved
( ) Defence departments; further distribution only as approved
( ) Other (please specify):

12 DOCUMENT ANNOUNCEMENT (any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availability (11) However, where further distribution (beyond the audience specified in (11) is possible, a wider announcement audience may be selected)

UNCLASSIFIED
SECURITY CLASSIFICATION OF FORM

UNCLASSIFIED
SECURITY CLASSIFICATION OF FORM
(highest classification of Title, Abstract, Keywords)

13. ABSTRACT (a brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), (R), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual)

The Ultimate Strength application is introduced. Its system architecture is described. A general overview of the representation of a cross section within the application is provided. Important aspects of the application are discussed. A bibliography and appendices are provided.

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus, e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus identified. If it not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title)

Ship structure
Ultimate strength
Hull girder strength
Longitudinal strength
Load shortening curve
Moment curvature
Stiffened panel

UNCLASSIFIED
SECURITY CLASSIFICATION OF FORM